Univerzita Karlova v Praze

Matematicko-fyzikální fakulta

HABILITAČNÍ PRÁCE

komentovaný soubor publikací

OBOR: Teoretická informatika

Minimální KNF reprezentace booleovských funkcí

RNDr. Petr Kučera, Ph.D.

Katedra teoretické informatiky a matematické logiky

Praha, říjen 2013

Obsah habilitační práce

Tato práce se skládá ze souboru pěti původních článků doplněného úvodním komentářem.

1. Endre Boros, Ondřej Čepek, Alexander Kogan, and Petr Kučera. Exclusive and essential sets of implicates of Boolean functions. **Discrete Applied Mathematics** 158 (2010) 81-96. (16 stran, [8])

2. Endre Boros, Ondřej Čepek, Alexander Kogan, and Petr Kučera. A subclass of Horn CNFs optimally compressible in polynomial time. **Annals of Mathematics and Artificial Intelligence** (2009) 57:249-291. (43 stran, [7])

3. Ondřej Čepek and Petr Kučera. Disjoint essential sets of implicates of a CQ Horn function. **Annals of Mathematics and Artificial Intelligence** (2011) 61:231-244. (14 stran, [12])

4. Ondřej Čepek, Petr Kučera, and Petr Savický. Boolean functions with a simple certificate for CNF complexity. **Discrete Applied Mathematics** 160 (2012) 365-382. (18 stran, [13])

5. Endre Boros, Ondřej Čepek, and Petr Kučera. A decomposition method for CNF minimality proofs. **Theoretical Computer Science**, *Available on-line*, DOI: 10.1016/j.bbr.2011.03.031, (16 stran, [9]).

Články do této práce byly vybrány s ohledem na společné téma, jímž je studium minimálních reprezentací booleovských funkcí formulemi v konjunktivně normální formě. Spojující téma vybraných článků se dá popsat i úžeji. Všechny se totiž nějakým způsobem věnují esenciálním a exkluzivním množinám implikátů, což jsou pojmy zavedené v prvním vybraném článku spolu s řadou teoretických výsledků o těchto množinách. Ve druhém článku byla pak takto vybudovaná teorie využita k definici nové třídy funkcí (component-wise quadratic, CQ), u nichž je možné v polynomiálním čase najít nejkratší reprezentaci formulí v konjunktivně normální formě. Ve třetím článku je třída CQ studována z hlediska esenciálních množin implikátů. Teorie z prvního článku je potom dále rozvíjena v článku čtvrtém. Konečně poslední vybraná práce popisuje způsob, jak pomocí esenciálních a exkluzivních množin implikátů odhadnout počet klauzulí v nejkratší reprezentaci dané funkce. Tyto techniky jsou použity k důkazu toho, že již hledání minimálních reprezentací formulí v 3KNF (tj. konjunktivně normální formě, kde každá klauzule obsahuje nejvýš tři literály) je NP-těžké. Všech pět článků již vyšlo v časopisech s impakt faktorem, přičemž poslední vybraný článek, tedy [9] vyšel zatím jen elektronicky a v době podání této habilitační práce jej ještě čeká přiřazení plných bibliografických údajů.

Podrobnější seznámení s výsledky obsaženými ve shromážděných článcích a související komentáře uvedeme v části dvě této habilitační práce. Nejprve však v první části práce shrneme současný stav problematiky minimalizace booleovských formulí, což je téma spojující vybrané články.

1 Současný stav zkoumané problematiky

Pod pojmem booleovská funkce na n proměnných míníme zobrazení $f : \{0,1\}^n \mapsto \{0,1\}$. Booleovskou funkci lze reprezentovat různými způsoby, asi nejpřirozenějším je reprezentace pomocí pravdivostní tabulky. Tento způsob reprezentace však jistě není nejúspornější.

Hledání v nějakém smyslu nejmenší či nejúspornější reprezentace dané booleovské funkce je přitom důležitým problémem, který je třeba řešit v řadě aplikací. Například v umělé inteligenci je tento problém ekvivalentní hledání nejúspornější reprezentace báze znalostí [20, 19]. Při tom dochází ke kompresi znalostí, neboť zatímco reprezentace se může změnit, samotné znalosti zůstávají beze změny. Při návrhu logických obvodů pak hledání nejmenší reprezentace odpovídá hledání obvodu s co nejmenším počtem hradel [10].

Obtížnost hledání úsporné reprezentace i způsoby, které k tomuto hledání bývají použity, se liší podle toho, jaký typ reprezentace uvažujeme. Roli hraje pochopitelně i to, jakým způsobem měříme velikost reprezentace, tedy použitá míra. Ve většině případů je problém minimalizace booleovské funkce obtížný, a proto se v praktických aplikacích často využívají heuristické metody. Již v šedesátých letech byly používány Karnaughovy mapy, algoritmus Espresso a řada jiných [10].

V závislosti na aplikaci se uvažují různé reprezentace booleovských funkcí. Zmiňme například rozhodovací diagramy (binary decision diagrams, BDD), OBDD (ordered BDD, jde o zvláštní typ rozhodovacího diagramu) či obvody. Obvod lze definovat jako orientovaný acyklický graf, v němž vnitřní vrcholy odpovídají hradlům, která počítají elementární booleovské funkce (např. konjunkci, disjunkci, negaci, ale i XOR či NAND). Vrcholy bez vstupních hran pak odpovídají vstupním proměnným a vrcholy bez výstupních hran odpovídají výstupu obvodu. Zvláštním případem obvodu je potom logická formule, na kterou lze pohlížet jako na obvod, u nějž vyžadujeme, aby výstupní stupeň každého hradla byl nejvýš jedna.

V mnoha případech uvažujeme reprezentaci booleovské funkce pomocí konjunktivně normální formy (KNF) či disjunktivně normální formy (DNF, též zvané *sum of products, SOP*). Je známo, že každou booleovskou funkci lze reprezentovat jak pomocí KNF tak pomocí DNF [11]. Z hlediska struktury jsou si tyto reprezentace velmi podobné, a proto obvykle stačí uvažovat jednu z nich, přičemž výsledky pro KNF lze snadno přenést i na DNF a naopak. V následujícím textu se proto omezíme zejména na KNF, protože i články obsažené v této práci používají tento typ reprezentace booleovských funkcí. Konjunktivně normální formu definujeme následujícím způsobem. *Literál* je buď proměnná (*pozitivní literál*, např. a), nebo její negace (*negativní literál*, např. $\overline{a}$). *Klauzule* je disjunkcí literálů, např. $(\overline{a} \vee b \vee \overline{c})$. Řekneme, že formule φ je v *konjunktivně normální formě* (KNF), jde-li o konjunkci klauzulí, např. $(\overline{a} \vee b \vee \overline{c})(a \vee \overline{b} \vee c)$. Místo toho, abychom psali, že φ je formulí v KNF, budeme též psát, že φ je KNF. Obsahuje-li KNF φ jen klauzule s nejvýš třemi literály, řekneme, že jde o 3KNF či že je φ *kubická*. Je-li KNF φ složena jen z klauzulí obsahujících nejvýš dva literály, řekneme, že jde o 2KNF či že je φ *kvadratická*.

Formule v KNF odpovídá obvodu, který má dvě vrstvy hradel, první vrstva obsahuje jen hradla typu OR (disjunkce), druhá vrstva pak hradla AND (konjunkce). Pokud připustíme, že jedno hradlo může mít více než dva vstupy, jde skutečně o obvod hloubky dva (kde hloubku obvodu definujeme jako počet hradel na nejdelší cestě od vstupu k výstupu obvodu), tedy o obvod, jehož paralelní výpočet zabere dva kroky. Pokud však vyžadujeme, aby hradla AND a OR měla vždy jen dva vstupy, je hloubka obecné KNF $\Theta(\log_2 n)$. I v tomto případě však hloubka formule v 3KNF nebo v 2KNF bude konstantní, což dodává studiu konjunktivně normálních forem s klauzulemi omezené délky na důležitosti.

Uveďme ještě několik základních pojmů, které budeme v tomto textu využívat. Nechť f je booleovská funkce. Řekneme, že klauzule C je *implikátem* funkce f, pokud platí, že každé ohodnocení splňující f splňuje i C. Řekneme, že implikát C funkce f je *primární*, pokud neexistuje žádná vlastní podklauzule C' klauzule C, která by též byla implikátem f. Formule φ v KNF reprezentující funkci f je *primární KNF*, pokud se skládá pouze z primárních implikátů f. Pokud z φ nelze odstranit žádnou klauzuli aniž by došlo ke změně reprezentované funkce, jedná se o *iredundantní KNF*.

Uvažme nejprve úlohu, jejímž vstupem je pravdivostní tabulka funkce f a na výstupu očekáváme formuli φ v KNF, která reprezentuje funkci f a skládá se z co nejmenšího počtu klauzulí (označme si tuto úlohu jako BM-TT). Takto definovaná úloha je zvláštním případem problému Set Cover, tedy pokrývání množin. Konkrétně se snažíme pokrýt *nulové body* (*false pointy*), což jsou vstupy v, pro něž je $f(v) = 0$, primárními implikáty funkce f. Stejně jako Set Cover [18], i BM-TT je NP-těžká úloha [1, 14, 27]. Navíc lze tuto úlohu těžko aproximovat. Konkrétně, pokud NP nepatří do třídy problémů řešitelných v deterministickém kvazipolynomiálním čase, pak existuje konstanta $\gamma > 0$, pro kterou platí, že neexistuje polynomiální aproximační algoritmus s aproximačním poměrem $(\log N)^\gamma$, kde N je velikost pravdivostní tabulky funkce f na vstupu [1, 17].

Úlohu, kdy uvažujeme jako vstup KNF φ reprezentující funkci f na n proměnných a cílem je nalézt KNF φ', která rovněž reprezentuje funkci f a skládá se z co nejmenšího počtu klauzulí, si označíme jako BM-KNF. Tato úloha je dokonce Σ_2^p-těžká, kde Σ_2^p označuje druhou úroveň polynomiální hierarchie. Nejen to, ale Σ_2^p-těžké je i aproximovat řešení úlohy BM-KNF v polynomiálním čase s aproximačním poměrem $n^{\frac{1}{2}-\varepsilon}$ pro libovolné $\varepsilon > 0$ [26, 27].

Kromě hledání minimálních reprezentací obecných booleovských funkcí se úsilí zaměřovalo i na studium zvláštních tříd booleovských funkcí. Mezi nimi význačné postavení zaujímá třída hornovských funkcí. Řekneme, že klauzule je *hornovská*, obsahuje-li nejvýš jeden pozitivní literál, přitom pokud obsahuje právě jeden pozitivní literál, hovoříme o *čistě hornovské klauzuli*, pokud pozitivní literál není v klauzuli vůbec neobsažen, jedná se o *negativní klauzuli*. *Hornovská KNF* se potom skládá jen z hornovských klauzulí. Pokud funkci f můžeme reprezentovat nějakou hornovskou KNF, hovoříme o *hornovské funkci*. Důležitost třídy hornovských formulí tkví jednak v tom, že o hornovské formuli φ můžeme v lineárním čase rozhodnout, zda je splnitelná [16, 23, 25], přestože problém testování splnitelnosti obecné formule v KNF je NP-úplný [18]. Dalším důvodem význačnosti hornovských formulí je, že analogické struktury se objevují v řadě oblastí diskrétní matematiky a informatiky. Mezi jinými zmiňme například orientované hypergrafy (viz např. [3]), databázová schémata [24, 15], implikační báze uzávěrových systémů [2, 4].

Na rozdíl od splnitelnosti, minimalizace hornovských funkcí vzhledem k počtu klauzulí je NP-těžkou úlohou, pokud je vstupem hornovská KNF [3] (takto omezený problém nazveme BM-HORN). Nedávno bylo navíc ukázáno, že jde o problém těžký i pro aproximaci. Konkrétně v [5] autoři ukazují, že problém BM-HORN není aproximovatelný v polynomiálním čase s poměrem $2^{\log^{1-\varepsilon}(n)}$ pro každé $\varepsilon > 0$ za předpokladu, že $NP \not\subseteq DTIME(n^{polylog(n)})$. Jen o něco novější výsledek [6] ukazuje, že problém BM-HORN je neaproximovatelný polynomiálním algoritmem s aproximačním poměrem $2^{O(\log^{1-o(1)} n)}$ za předpokladu, že $P \neq NP$, a to i tehdy, je-li vstup omezen na hornovské formule v 3KNF.

Na druhou stranu však zmiňme i některé pozitivní výsledky týkající se minimalizace hornovských funkcí. Čistě hornovskou klauzuli C můžeme rozdělit na dvě části, na množinu proměnných A, které se v C vyskytují negativně a na proměnnou x, která se v C vyskytuje pozitivně. V tom případě říkáme, že A je *zdrojová množina* a proměnné v ní obsažené jsou *podcíle* klauzule C. Jediná pozitivní proměnná x je pak *cílem* klauzule C. Tyto pojmy vycházejí z toho, že klauzule $C = (\bigvee_{a\in A} \overline{a} \vee x)$ reprezentuje implikaci $(\bigwedge_{a\in A} a \rightarrow x)$, kterou si můžeme představit jako pravidlo, které umožňuje z platnosti podcílů odvodit platnost cíle klauzule C. Úloha, v níž je naším cílem nalézt pro danou hornovskou KNF φ formuli φ' s co nejmenším počtem různých zdrojových množin, je řešitelná v polynomiálním čase [24].

Kromě toho můžeme nalézt zvláštní případy hornovských funkcí, pro něž je možné vyřešit úlohu BM-HORN v polynomiálním čase. Jedná se například o třídy acyklických a kvaziacyklických funkcí zavedených v [21]. Definice obou těchto tříd je založena na pojmu KNF grafu. K dané hornovské KNF φ definujeme KNF graf $G_\varphi = (V, E)$ následovně, jde o orientovaný graf, kde množina vrcholů V odpovídá množině proměnných φ. Z proměnné x do proměnné y vede hrana, tj. $(x, y) \in E$, pokud ve formuli φ existuje klauzule C, v níž je x podcílem a y cílem. Dá se ukázat, že definují-li formule φ a φ' touž funkci f, pak grafy G_φ a $G_{\varphi'}$ mají shodný tranzitivní uzávěr G_f, naopak to ovšem neplatí. *Acyklická funkce* f má acyklický graf G_f. Pro *kvaziacyklickou funkci* g platí, že patří-li dvě proměnné x a y do téže silně souvislé komponenty grafu G_g, pak jsou pro funkci g proměnné x a y ekvivalentní, tj. $(\overline{x} \vee y)$ a $(\overline{y} \vee x)$ jsou implikáty funkce g. V [21] se ukazuje, že každá primární KNF φ reprezentující acyklickou (resp. kvaziacyklickou) funkci f je acyklická (resp. kvaziacyklická). Navíc je možno najít k ní nejkratší KNF reprezentaci φ' funkce f v polynomiálním čase. nalezená KNF φ' je potom nejkratší jak co do počtu klauzulí, tak co do celkového počtu literálů, které se ve formuli φ' vyskytují.

V obecném případě lze problém BM-HORN aproximovat v polynomiálním čase s aproximačním poměrem n, kde n označuje počet proměnných vstupní formule φ. K tomu stačí převést hornovskou KNF φ do primární a iredundantní podoby [20]. Je známo [20], že k dané hornovské formuli φ lze nalézt ekvivalentní primární a iredundantní formuli v polynomiálním čase.

Za zmínku také stojí, že dosud není známa složitost hledání minimální hornovské KNF reprezentující funkci f, je-li zadána pravdivostní tabulkou.

2 Přehled prezentovaných výsledků

V článku [8] jsou zavedeny ústřední pojmy esenciální a exkluzivní množiny implikátů, které jsou dále využívány v dalších článcích. Obě definice jsou založené na pojmech rezoluce a rezolučního odvození, které nejprve krátce připomeneme. Jsou-li C_1 a C_2 klauzule, řekneme, že mají *konflikt v proměnné* x, pokud jedna z nich obsahuje x a druhá $\overline{x}$. Pokud C_1 a C_2 mají konflikt v právě jedné proměnné, pak jsou *rezolvovatelné*. Jsou-li $C_1 = A \vee x$ a $C_2 = B \vee \overline{x}$, kde A a B jsou klauzule, které nemají konflikt v žádné proměnné, potom *rezolventa* klauzulí C_1 a C_2 je klauzule $C = R(C_1, C_2) = (A \vee B)$. Dá se ukázat, že je-li φ formule v KNF, která reprezentuje funkci f, a lze-li klauzuli C odvodit z formule φ posloupností rezolucí, pak C je implikát f. Navíc platí, že každý primární implikát funkce f lze z φ pomocí rezoluce odvodit [11].

Nechť f je booleovská funkce, pomocí $\mathcal{I}^p(f)$ označíme množinu všech primárních implikátů f a pomocí $\mathcal{I}(f)$ označíme rezoluční uzávěr $\mathcal{I}^p(f)$, tj. do $\mathcal{I}(f)$ patří všechny klauzule, které lze odvodit z primárních implikátů f pomocí rezoluce.

Nyní již můžeme definovat pojmy esenciální a exkluzivní množiny implikátů. Nechť f je booleovská funkce a nechť $\mathcal{X} \subseteq \mathcal{I}(f)$, řekneme, že $\mathcal{X}$ je *exkluzivní* množinou implikátů funkce f, pokud pro každé dvě rezolvovatelné klauzule $C_1, C_2 \in \mathcal{I}(f)$ platí, že je-li jejich rezolventa $C = R(C_1, C_2)$ prvkem $\mathcal{X}$, pak i oba implikáty C_1, C_2 patří do $\mathcal{X}$. Tj. platí-li implikace

$$C = R(C_1, C_2) \in \mathcal{X} \Rightarrow C_1 \in \mathcal{X} \text{ a } C_2 \in \mathcal{X}.$$

Podobně definujeme i esenciální množiny. Množina $\mathcal{E} \subseteq \mathcal{I}(f)$ je *esenciální* množinou implikátů funkce f, pokud pro každé dvě rezolvovatelné klauzule $C_1, C_2 \in \mathcal{I}(f)$ platí, že je-li jejich rezolventa $C = R(C_1, C_2)$ prvkem $\mathcal{E}$, pak alespoň jeden z implikátů C_1, C_2 patří rovněž do $\mathcal{E}$. Tj. platí-li implikace

$$C = R(C_1, C_2) \in \mathcal{X} \Rightarrow C_1 \in \mathcal{E} \text{ nebo } C_2 \in \mathcal{E}.$$

V článku [8] je studována řada vlastností těchto množin, zmiňme zde ty nejzajímavější. Význam exkluzivních množin pro minimalizaci booleovských funkcí tkví v následující výměnné vlastnosti. Uvažme booleovskou funkci f a exkluzivní množinu implikátů $\mathcal{X} \subseteq \mathcal{I}(f)$. V [8] se ukazuje, že jsou-li φ_1 a φ_2 dvě KNF formule reprezentující f, pak $\varphi_1 \cap \mathcal{X}$ reprezentuje touž funkci jako $\varphi_2 \cap \mathcal{X}$, takovou funkci pak nazýváme exkluzivní komponentou funkce f a označujeme ji pomocí $f_\mathcal{X}$. To znamená, že v KNF reprezentaci f můžeme zaměňovat různé reprezentace exkluzivní komponenty $f_\mathcal{X}$. Při hledání nejmenší reprezentace f můžeme tedy zvlášť nalézt nejmenší reprezentaci komponenty $f_\mathcal{X}$, kterou poté vložíme zpět do KNF reprezentace f. Exkluzivní množiny implikátů funkce f takto umožňují dekomponovat problém minimalizace na podproblémy.

Zastavme se nyní u esenciálních množin implikátů. V [8] se ukazuje, že jsou-li $\mathcal{E}_1, \ldots, \mathcal{E}_k$ po dvou disjunktní esenciální množiny implikátů funkce f, pak každá KNF formule φ reprezentující f musí obsahovat alespoň k klauzulí. Tento dolní odhad plyne z obecnějšího tvrzení, které říká, že je-li φ formule v KNF, pak φ reprezentuje f, právě když φ má neprázdný průnik s každou neprázdnou esenciální množinou implikátů funkce f. Označíme-li si počet klauzulí v nejmenší (vzhledem k počtu klauzulí) KNF reprezentaci f pomocí $cnf(f)$ a největší počet po dvou disjunktních esenciálních množin implikátů funkce f pomocí $ess(f)$, dostáváme nerovnost $ess(f) \leq cnf(f)$. Esenciální množiny takto mohou poskytnout dolní odhad na počet klauzulí v nejkratší KNF (co do počtu klauzulí) reprezentující danou funkci f.

V [8] jsou studovány i další vlastnosti esenciálních a exkluzivních množin, a to jak obecných booleovských funkcí, tak zvláště hornovských funkcí a kvadratických funkcí. Ukazuje se, že je-li funkce f hornovská kvaziacyklická funkce či je-li f kvadratická funkce, pak platí dokonce rovnost $ess(f) = cnf(f)$. To nabízí další vhled do principu práce minimalizačních algoritmů pro kvaziacyklické (a tedy i acyklické) a kvadratické funkce.

V článku [7] jsou teoretické výsledky z [8] využity k zobecnění kvaziacyklických funkcí do třídy CQ funkcí a ke konstrukci polynomiálního algoritmu, který nalezne nejkratší reprezentaci CQ funkce. Připomeňme, že definice kvaziacyklické funkce f je založena na KNF grafu G_f. Tento graf je i základem definice CQ funkce. Uvažme opět hornovskou funkci f. Řekneme, že čistě hornovská klauzule C je *CQ klauzulí* (*component-wise quadratic*) vzhledem k funkci f, pokud platí, že je-li K silně souvislá komponenta grafu G_f obsahující hlavu klauzule C, pak K obsahuje nejvýš jeden podcíl C. Je-li φ KNF reprezentující f, která obsahuje pouze CQ implikáty a f, pak φ nazveme *CQ KNF reprezentací* f. Pokud f má alespoň jednu CQ KNF reprezentaci, jde o *CQ funkci*. Dá se přitom ukázat, že je-li f CQ funkce a C její primární implikát, tak C je CQ klauzulí vzhledem k f. Z toho plyne, že každá primární KNF reprezentace f je současně CQ KNF reprezentací f. Všimněme si, že z uvedené definice plyne, že primární implikáty funkce f jsou čistě hornovské, speciálně nepřipouštíme negativní implikáty. Již z výsledků v [21] plyne, že u hornovské funkce f lze zvlášť minimalizovat čistě hornovskou část, platí, že v každé minimální KNF reprezentaci f je potom týž počet negativních implikátů a že je možné minimalizovat počet negativních implikátů s jakoukoli čistě hornovskou částí. To plyne i z toho, že čistě hornovské implikáty tvoří exkluzivní množinu, zatímco negativní implikáty tvoří esenciální množinu implikátů funkce f.

CQ funkce jsou zřejmě zobecněním kvaziacyklických funkcí. Poznamenejme, že kdybychom připustili dva podcíle v téže silně souvislé komponentě společně s hlavou, patřily by do takto zobecněné třídy všechny hornovské 3KNF, a pro ně již je minimalizace NP-úplná [9].

Minimalizační algoritmus v [7] je založen na rozložení do exkluzivních komponent, které jsou definovány na základě nově zavedeného grafu D_f implikátů dané hornovské funkce f. K minimalizaci samé je pak využito KNF grafu G_f. Algoritmus nalezne v polynomiálním čase nejmenší reprezentaci dané CQ funkce f jak vzhledem k počtu klauzulí, tak vzhledem k počtu literálů. Vlastnosti třídy CQ funkcí jsou dále studovány v [12], kde se ukazuje, že i v případě CQ funkce f vždy platí, že $cnf(f) = ess(f)$.

Funkce, pro něž platí rovnost mezi mírami $cnf(f)$ a $ess(f)$ jsou podrobeny systematickému studiu v článku [13]. Funkce f, která splňuje $cnf(f) = ess(f)$ je zde nazvána *coverable*. Nejprve jsou v [13] zkoumány další vlastnosti esenciálních množin. Předně se ukazuje, že při určování míry $ess(f)$ se stačí omezit na esenciální množiny definované pomocí nulových bodů dané funkce. Je-li v nulovým bodem funkce f, tj. platí-li $f(v) = 0$, pak množina $\mathcal{E}(v) = \{C \in \mathcal{I}(f) \mid C(v) = 0\}$ je esenciální množinou. Všimněme si, že například esenciální množina všech negativních implikátů hornovské funkce f, má-li f jaké, je takto definována pomocí nulového bodu v, který obsahuje samé jedničky. V [8] bylo těchto množin využito při důkazu jednoho směru ekvivalence, že KNF φ reprezentuje funkci f právě když má neprázdný průnik s každou esenciální množinou implikátů funkce f.

V [13] jsou tyto množiny nazvány *FE množinami* (*falsepoint essential set*) a je zde ukázáno, že zmíněná ekvivalence platí již pro případ FE množin. Navíc se ukazuje, že v inkluzi minimální esenciální množina je vždy FE množinou a i pro definici $ess(f)$ si ve skutečnosti vystačíme s FE množinami. Ve skutečnosti se stačí omezit jen na průniky esenciálních množin s $\mathcal{I}^p(f)$, tj. jen na primární implikáty z FE množin. U dvou FE množin $\mathcal{E}(u)$ a $\mathcal{E}(v)$ není obtížné rozhodnout, zda jsou disjunktní, což je základem důkazu toho, že problém, v němž se ptáme, zda $ess(f) \geq k$ pro danou funkci f a přirozené číslo k, patří do NP. Na druhou stranu se v článku [13] ukazuje i NP-těžkost tohoto problému, jde tedy o problém NP-úplný. Zvláštním případem jsou potom coverable funkce. Uvažme třídu KNF formulí $\mathcal{C}$, která je coverable a *tractable*, což v zásadě znamená, že je možno v polynomiálním čase rozhodnout, zda daná KNF φ patří do $\mathcal{C}$, zda daná KNF $\varphi \in \mathcal{C}$ je splnitelná a zda daná klauzule je implikátem dané KNF $\varphi \in \mathcal{C}$. Z hlediska splnitelnosti je tedy tractable třída formulí jednoduchá. Pro takovou třídu $\mathcal{C}$ platí, že problém, v němž se ptáme, zda $cnf(f) \leq k$ pro danou funkci f reprezentovanou KNF formulí $\varphi \in \mathcal{C}$ a dané přirozené číslo k, patří do $NP \cap coNP$.

Zdaleka ne všechny funkce jsou ovšem coverable. V [13] je studována i otázka, jak velký může být rozdíl mezi $cnf(f)$ a $ess(f)$ pro obecnou funkci f, případně pro hornovskou funkci f. Ukazuje se, že poměr $cnf(f)/ess(f)$ nelze omezit žádnou konstantou, přesněji je zkonstruován příklad hornovské funkce na n proměnných, pro kterou je tento poměr $\Omega(\log_2 n)$. Možná velikost tohoto poměru byla dále studována autorkami [22], které zkonstruovaly hornovskou funkci f, pro kterou je $cnf(f)/ess(f) = \Omega(\sqrt{n})$.

Konečně v článku [9] je popsána obecná metoda, která s pomocí dekompozice implikátů booleovské funkce f slouží k určení dolního odhadu počtu klauzulí v nejkratší KNF reprezentující f, tedy dolního odhadu hodnoty $cnf(f)$. Tato metoda je demonstrována na dvou příkladech. Jedním z nich je důkaz NP-těžkosti minimalizace hornovských funkcí, tento důkaz původně pochází z [3], avšak tam byl důkaz jen velmi zkratkovitý. V [9] je popsaná dekompoziční metoda použita k elegantnímu a jednoduchému důkazu potřebných vlastností převodu popsaného v [3]. Tento důkaz je posléze upraven tak, aby ukazoval NP-těžkost minimalizace hornovských 3KNF, i zde dekompoziční metoda popsaná v [9] podává jednoduchý nástroj pro důkaz minimality formule zkonstruované v převodu z problému Set Cover. Uvědomme si, že prostý fakt, že minimalizace hornovských 3KNF je NP-těžká vyplývá již z výsledků [6], hlavním výsledkem [9] je popis metody, která umožňuje ukazovat dolní odhad $cnf(f)$ pro konkrétní funkci f, minimalizace hornovských funkcí je zde vzata především jako příklad, na kterém jsou techniky popsané v [9] demonstrovány.

Reference

[1] Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael Saks. Minimizing dnf formulas and ac_d^0 circuits given a truth table. In *Proceedings of the 21st Annual IEEE Conference on Computational Complexity*, pages 237–251. IEEE Computer Society, 2006. ISBN 0-7695-2596-2. doi: 10.1109/CCC.2006.27.

[2] Marta Arias and José Balcázar. Canonical horn representations and query learning. In Ricard Gavaldà, Gábor Lugosi, Thomas Zeugmann, and Sandra Zilles, editors, *Algorithmic Learning Theory*, volume 5809 of *Lecture Notes in Computer Science*, pages 156–170. Springer Berlin / Heidelberg, 2009. ISBN 978-3-642-04413-7.

[3] G. Ausiello, A. D'Atri, and D. Sacca. Minimal representation of directed hypergraphs. *SIAM Journal on Computing*, 15(2):418–431, May 1986.

[4] K. Bertet and B. Monjardet. The multiple facets of the canonical direct unit implicational basis. *Theoretical Computer Science*, 411(22–24):2155 – 2166, 2010. ISSN 0304-3975. doi: 10.1016/j.tcs.2009.12.021.

[5] Amitava Bhattacharya, Bhaskar DasGupta, Dhruv Mubayi, György Turán, Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul Spirakis. *On Approximate Horn Formula Minimization*, volume 6198, pages 438–450. Springer Berlin / Heidelberg, 2010. ISBN 978-3-642-14164-5. doi: 10.1007/978-3-642-14165-2_38.

[6] Endre Boros and Aritanan Gruber. Hardness results for approximate pure horn cnf formulae minimization. In *Proceedings of International Symposium on AI and Mathematics (ISAIM)*, 2012.

[7] Endre Boros, Ondřej Čepek, Alexander Kogan, and Petr Kučera. A subclass of horn cnfs optimally compressible in polynomial time. *Annals of Mathematics and Artificial Intelligence*, 57:249–291, 2009. ISSN 1012-2443. 10.1007/s10472-010-9197-7.

[8] Endre Boros, Ondřej Čepek, Alexander Kogan, and Petr Kučera. Exclusive and essential sets of implicates of boolean functions. *Discrete Applied Mathematics*, 158(2):81 – 96, 2010. ISSN 0166-218X. doi: 10.1016/j.dam.2009.08.012.

[9] Endre Boros, Ondřej Čepek, and Petr Kučera. A decomposition method for cnf minimality proofs. *Theoretical Computer Science*, 2013. ISSN 0304-3975. doi: 10.1016/j.tcs.2013.09.016.

[10] R.K. Brayton, G.D. Hachtel, C. McMullen, and A.L. Sangiovanni-Vincentelli. *Logic minimization algorithms for VLSI synthesis*, volume 2. Springer, 1984.

[11] Hans K. Büning and T. Lettmann. *Propositional Logic: Deduction and Algorithms*. Cambridge University Press, New York, NY, USA, 1999. ISBN 0521630177.

[12] Ondřej Čepek and Petr Kučera. Disjoint essential sets of implicates of a cq horn function. *Annals of Mathematics and Artificial Intelligence*, 61:231–244, 2011. ISSN 1012-2443. doi: 10.1007/s10472-011-9263-9.

[13] Ondřej Čepek, Petr Kučera, and Petr Savický. Boolean functions with a simple certificate for cnf complexity. *Discrete Applied Mathematics*, 160(4-5):365 – 382, March 2012. ISSN 0166-218X.

[14] Sebastian Czort. The complexity of minimizing disjunctive normal form formulas. Master's thesis, University of Aarhus, 1999.

[15] C. Delobel and R.G. Casey. Decomposition of a data base and the theory of boolean switching functions. *IBM Journal of Research and Development*, 17:374 – 386, 1973.

[16] W.F. Dowling and J.H. Gallier. Linear time algorithms for testing the satisfiability of propositional horn formulae. *Journal of Logic Programming*, 3:267 – 284, 1984.

[17] Vitaly Feldman. Hardness of approximate two-level logic minimization and pac learning with membership queries. In *STOC '06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 363–372, New York, NY, USA, 2006. ACM. ISBN 1-59593-134-1. doi: http://doi.acm.org/10.1145/1132516.1132569.

[18] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, San Francisco, 1979.

[19] P. L. Hammer and A. Kogan. Knowledge compression - logic minimization for expert systems. In *Proceedings of IISF/ACM Japan International Symposium*, pages 306–312, Tokyo, March 1994. World Scientific, Singapore.

[20] P.L. Hammer and A. Kogan. Optimal compression of propositional horn knowledge bases: Complexity and approximation. *Artificial Intelligence*, 64:131 – 145, 1993.

[21] P.L. Hammer and A. Kogan. Quasi-acyclic propositional horn knowledge bases: Optimal compression. *IEEE Transactions on Knowledge and Data Engineering*, 7(5):751 – 762, 1995.

[22] Lisa Hellerstein and Devorah Kletenik. On the gap between ess(f) and cnf_size(f). *Discrete Applied Mathematics*, 161(1-2):19 – 27, 2013. ISSN 0166-218X. doi: 10.1016/j.dam.2012.07.004.

[23] A. Itai and J.A. Makowsky. Unification as a complexity measure for logic programming. *Journal of Logic Programming*, 4:105 – 117, 1987.

[24] D. Maier. Minimal covers in the relational database model. *Journal of the ACM*, 27:664 – 674, 1980.

[25] M. Minoux. Ltur: A simplified linear time unit resolution algorithm for horn formulae and computer implementation. *Information Processing Letters*, 29:1 – 12, 1988.

[26] Christopher Umans. The minimum equivalent dnf problem and shortest implicants. *J. Comput. Syst. Sci.*, 63(4):597–611, 2001. ISSN 0022-0000. doi: http://dx.doi.org/10.1006/jcss.2001.1775.

[27] Christopher Umans, Tiziano Villa, and Alberto L. Sangiovanni-Vincentelli. Complexity of two-level logic minimization. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 25(7):1230–1246, 2006.

Discrete Applied Mathematics 158 (2010) 81–96

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Exclusive and essential sets of implicates of Boolean functions

Endre Boros [a], Ondřej Čepek [b,c,*], Alexander Kogan [d,a], Petr Kučera [b]

[a] *RUTCOR, Rutgers University, P.O. Box 5062, New Brunswick, NJ 08903, USA*

[b] *Department of Theoretical Computer Science, Charles University, Malostranské nám. 25, 118 00 Praha 1, Czech Republic*

[c] *Institute of Finance and Administration (VŠFS), Estonská 500, 100 00 Praha 10, Czech Republic*

[d] *Department of Accounting, Business Ethics, and Information Systems, Rutgers Business School, Rutgers University, Newark, NJ 07102, USA*

A R T I C L E   I N F O

Article history:
Received 25 June 2008
Received in revised form 25 July 2009
Accepted 27 August 2009
Available online 30 September 2009

Keywords:
Horn functions
Boolean minimization
Essential sets
Exclusive sets

A B S T R A C T

In this paper we study relationships between CNF representations of a given Boolean function f and certain sets of implicates of f. We introduce two definitions of sets of implicates which are both based on the properties of resolution. The first type of sets, called exclusive sets of implicates, is shown to have a functional property useful for decompositions. The second type of sets, called essential sets of implicates, is proved to possess an orthogonality property, which implies that every CNF representation and every essential set must intersect. The latter property then leads to an interesting question, to which we give an affirmative answer for some special subclasses of Horn Boolean functions.

1. Introduction

One of the most commonly used representations of Boolean functions are CNFs (conjunctive normal forms). For a given function there are typically many different CNFs representing it, which may significantly vary in length. In some applications an important problem is the following: for a given function find a shortest CNF among all of its possible CNF representations. For instance, in artificial intelligence this problem is equivalent to finding a most compact representation of a given knowledge base [12,13]. Such transformation of a knowledge base accomplishes knowledge compression, since the actual knowledge does not change, while the size of the representation can be significantly reduced. In general, this problem, known as Boolean minimization (BM), can be stated as follows: given a CNF ϕ find a CNF ϕ' representing the same function and such that ϕ' consists of a minimum possible number of clauses.

It is easy to see that BM is NP-hard as it contains the satisfiability problem (SAT) as its special case (an unsatisfiable CNF can be represented by an equivalent CNF consisting of only the constant "false", i.e. consisting of zero clauses or one clause depending on whether the definition of a clause admits clauses with no variables). In fact, BM was shown to be probably harder than SAT: while SAT is NP-complete (i.e. Σ_1^p-complete) [6], BM is Σ_2^p-complete [21] (see also the review paper [22] for related results). BM remains NP-hard even for some classes of Boolean functions for which SAT is solvable in polynomial time. The best known example of such a class are Horn functions (see [1,2,7,12,15] for various BM intractability results). The difficulty of BM of course raises a natural question whether for a given input CNF a nontrivial lower bound can be obtained for the number of clauses in the shortest equivalent CNF. We give a partial answer to this question in Section 6.

[*] Corresponding author at: Department of Theoretical Computer Science, Charles University, Malostranskénám. 25, 118 00 Praha 1, Czech Republic. Tel.: +420 221 914 246; fax: +420 221 914 323.

E-mail addresses: boros@rutcor.rutgers.edu (E. Boros), cepek@rutcor.rutgers.edu, cepek@ksi.ms.mff.cuni.cz (O. Čepek), kogan@rutgers.edu (A. Kogan), kucerap@ktiml.ms.mff.cuni.cz (P. Kučera).

0166-218X/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2009.08.012

On the positive side, [14] introduced two subclasses of Horn functions, acyclic and quasi-acyclic functions, for which BM is solvable in polynomial time. In fact, [14] and an earlier paper [10] by the same authors served as the original motivation for the results presented in this manuscript.

Given a Boolean function f the main object of interest throughout this paper will be the set $\mathcal{I}(f)$ defined as the resolution closure of the set of all prime implicates of f. We shall show that $\mathcal{I}(f)$ has an interesting inner structure. We shall define and study certain subsets of $\mathcal{I}(f)$ called exclusive and essential sets of implicates of f, and show how these sets can be used for decompositions of f and for lower bounds on the size of CNF representations of f. Among other results we shall generalize the key theorems from [10,14] used to prove the correctness of the BM algorithm for quasi-acyclic Horn functions.

Our interest in the structure of $\mathcal{I}(f)$ was originally motivated by our attempts to design a polynomial time BM algorithm for a class of CQ Horn functions which form a strict superset of quasi-acyclic Horn functions. This algorithm was recently published in [4]. However, in the process of proving the correctness of the above mentioned algorithm, we have developed many nontrivial results concerning the structure of $\mathcal{I}(f)$ and its connections to the properties of CNF representations of f, which are not restricted to the CQ Horn or even general Horn cases, but are valid for all Boolean functions. Therefore, we believe that these results are of an independent interest, and present them in this paper, which is structured as follows.

In Section 2 we introduce the necessary notation and present several elementary results important for the subsequent presentation. In Section 3 we define the class of Horn functions as well as its subclasses of acyclic and quasi-acyclic functions and recall some basic properties of these classes. Section 4 contains a short collection of simple lemmas about the properties of resolution closures. The main results of this paper are presented in Sections 5 and 6. Section 5 deals with exclusive sets of implicates and yields a decomposition theorem, which is shown to be useful in Boolean minimization. Section 6 is devoted to essential sets of implicates and presents a duality relation between essential sets of clauses and CNF representations. This duality provides a lower bound for the size of a CNF representation and leads to an interesting question for which classes of functions this bound is tight. Finally, Section 7 gives an affirmative answer to this question for the classes of quadratic, acyclic, and quasi-acyclic Horn functions.

2. Basic notation, definitions, and results

In this section we shall introduce the necessary notation and state several basic known results that will be needed later in the text.

A *Boolean function* f on n propositional variables $x_1, \ldots, x_n$ is a mapping $\{0,1\}^n \to \{0,1\}$. The propositional variables $x_1, \ldots, x_n$ and their negations $\bar{x}_1, \ldots, \bar{x}_n$ are called *literals* (*positive* and *negative literal*s, respectively). An elementary disjunction of literals

$$C = \bigvee_{i \in I} \bar{x}_i \vee \bigvee_{j \in J} x_j \tag{1}$$

is called a *clause*, if every propositional variable appears in it at most once, i.e. if $I \cap J = \emptyset$. A *degree* of a clause is the number of literals in it. For two Boolean functions f and g we write $f \le g$ if

$$\forall (x_1, \ldots, x_n) \in \{0,1\}^n : f(x_1, \ldots, x_n) = 1 \Longrightarrow g(x_1, \ldots, x_n) = 1. \tag{2}$$

Since each clause is in itself a Boolean function, formula (2) also defines the meaning of inequalities $C_1 \le C_2$, $C_1 \le f$, and $f \le C_1$, where C_1, C_2 are clauses and f is a Boolean function.

We say that a clause C_1 *subsumes* another clause C_2 if $C_1 \le C_2$ (e.g. the clause $\bar{x} \vee z$ subsumes the clause $\bar{x} \vee \bar{y} \vee z$). A clause C is called an *implicate* of a function f if $f \le C$. An implicate C is called *prime* if there is no distinct implicate C' subsuming C, or in other words, an implicate of a function is prime if dropping any literal from it produces a clause which is not an implicate of that function.

It is a well-known fact that every Boolean function f can be represented by a conjunction of clauses (see e.g. [9]). Such an expression is called a *conjunctive normal form* (or CNF) of the Boolean function f. It should be noted that a given Boolean function may have many CNF representations (doubly exponentially many in the number of propositional variables). If two distinct CNFs, say ϕ_1 and ϕ_2, represent the same function, we say that they are *equivalent*, and denote this fact by $\phi_1 \equiv \phi_2$. A CNF ϕ representing a function f is called *prime* if each clause of ϕ is a prime implicate of the function f. A CNF ϕ representing a function f is called *irredundant* if dropping any clause from ϕ produces a CNF that does not represent f.

Example 2.1. Consider the CNF

$$(\bar{x}_1 \vee x_2) \wedge (\bar{x}_1 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_4) \wedge (\bar{x}_2 \vee \bar{x}_3 \vee x_4) \wedge (\bar{x}_3 \vee \bar{x}_4).$$

The 2nd clause can be dropped (although it is prime), and the 4th clause can be shortened (i.e. it is not prime). In fact, the same function can be represented by the CNF

$$(\bar{x}_1 \vee x_2) \wedge (\bar{x}_1 \vee x_4) \wedge (\bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_3 \vee \bar{x}_4)$$

which is both prime and irredundant.

The following two notational conventions will allow us to switch back and forth between sets of clauses and CNFs. For an arbitrary set of clauses $\mathcal{C}$ the symbol $\phi(\mathcal{C})$ denotes the CNF obtained by taking a conjunction of all clauses in $\mathcal{C}$. On the other hand, for an arbitrary CNF ϕ the symbol $\mathcal{C}(\phi)$ denotes the set of all clauses present in ϕ. We shall use the notion of "representing a given function" interchangeably for both CNFs and sets of clauses, i.e. if a CNF ϕ represents a function f we shall also say that the set of clauses $\mathcal{C}(\phi)$ represents f.

It should be noted here that all the results presented in this paper using the CNF terminology can be easily translated into equivalent statements using the DNF (disjunctive normal form) language with proper terminological substitutions (terms instead of clauses, implicants instead of implicates, consensus instead of resolution, falsifiability instead of satisfiability, etc.).

Two clauses C_1 and C_2 are said to be *resolvable* if they contain exactly one complementary pair of literals, i.e. if there exists exactly one propositional variable that appears uncomplemented in one of the clauses and complemented in the other. That means that we can write $C_1 = \tilde{C}_1 \vee x$ and $C_2 = \tilde{C}_2 \vee \bar{x}$ for some propositional variable x and clauses $\tilde{C}_1$ and $\tilde{C}_2$ which contain no complementary pair of literals. The clauses C_1 and C_2 are called *parent clauses* and the disjunction $R(C_1, C_2) = \tilde{C}_1 \vee \tilde{C}_2$ is called the *resolvent* of the parent clauses C_1 and C_2. Note that the resolvent is a clause (does not contain a propositional variable and its negation). The following is an easy lemma [5].

Lemma 2.2. *Let C_1 and C_2 be two resolvable implicates of a Boolean function f. Then $R(C_1, C_2)$ is also an implicate of f.*

We say, that a clause C can be *derived by a series of resolutions* from a CNF ϕ, if there exists a finite sequence $C_1, C_2, \ldots, C_p$ of clauses such that

(1) $C_p = C$, and
(2) for $i = 1, \ldots, p$, either $C_i \in \mathcal{C}(\phi)$ or there exist $j < i$ and $k < i$ such that $C_i = R(C_j, C_k)$.

Resolutions have a very important property usually called the *completeness of resolution*. Sometimes this property is also referred to as the Quine theorem after the author of one of the first papers in which this property was proved [16,17], see also [5] for related material.

Theorem 2.3. *Let ϕ be a CNF representation of a Boolean function f and let C be a prime implicate of f. Then C can be derived from ϕ by a series of resolutions.*

Throughout this paper we shall also use the following notation. For an arbitrary set of clauses $\mathcal{C}$ the *resolution closure* of $\mathcal{C}$ denoted by $\mathcal{R}(\mathcal{C})$ is the set of all clauses obtainable through series of resolutions from the set $\mathcal{C}$ (allowing the resolvents to become parent clauses in subsequent resolutions).

For a Boolean function f let us denote by $\mathcal{I}^p(f)$ the set of its prime implicates, and let $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}^p(f))$. Note that not all implicates of f may belong to $\mathcal{I}(f)$. For instance, if f is defined by the CNF $\phi = (x_1 \vee x_2) \wedge (x_2 \vee x_3)$, then we have $\mathcal{I}(f) = \mathcal{I}^p(f) = \{(x_1 \vee x_2), (x_2 \vee x_3)\}$, however the clause $(x_1 \vee x_2 \vee x_3)$ is also a implicate of f.

Let us now turn our attention to a subclass of Boolean functions which will be frequently used in this paper, namely the class of Horn functions.

3. Horn functions and their subclasses

In this section we will describe a class of Horn functions, the properties of which served as the original motivation for deriving the more general results presented in this paper. We will also define two subclasses of Horn functions which are the main subject of the last section.

A clause C defined by (1) is called *negative* if it contains no positive literals (i.e. if $J = \emptyset$). It is called *pure Horn* (or in some literature *definite Horn*) if it contains exactly one positive literal (i.e. if $|J| = 1$). To simplify notation, we shall sometimes write a pure Horn clause $C = \bigvee_{x \in S} \bar{x} \vee y$ simply as $C = S \vee y$. Each propositional variable $x \in S$ is called a *subgoal of C* and the propositional variable y is called the *head of C*.[1] We shall denote $Head(C) = y$, $Subg(C) = S$ (this set is sometimes called the "body" of C), and $Vars(C) = S \cup \{y\}$.

A CNF is called *Horn* if it contains only negative and pure Horn clauses. A CNF is called *pure Horn* if it contains only pure Horn clauses. Finally, a Boolean function is called *Horn* if it has at least one representation by a Horn CNF, and similarly a Boolean function is called *pure Horn* if it has at least one representation by a pure Horn CNF.

It is known (see [10]) that each prime implicate of a Horn function is either negative or pure Horn, and each prime implicate of a pure Horn function is pure Horn. Thus, in particular, any prime CNF representing a Horn function is Horn, and any prime CNF representing a pure Horn function is pure Horn.

Let us now recall some very useful definitions from [14], associating directed graphs to Horn CNFs and Horn functions. Let us start by reviewing several standard notions from graph theory.

A *directed graph* (or a *digraph*) is an ordered pair $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ where $\mathbf{N}$ is the set of *nodes* and $\mathbf{A}$ is the set of *arcs*, where an arc is an ordered pair of nodes. A *directed path* is a sequence of arcs $a_1, a_2, \ldots, a_p$ such that $a_i = (x_i, x_{i+1})$ for some nodes $x_1, x_2, \ldots, x_{p+1}$. A *directed cycle* is a directed path such that $x_1 = x_{p+1}$. A directed graph is called *strongly connected* if for any two nodes x and y there exist both a directed path from x to y and a directed path from y to x. If a graph $\mathbf{D}$ is not strongly connected then its node set can be decomposed in a unique way into maximal strongly connected subsets, called the *strong components* of $\mathbf{D}$. A directed graph is called *acyclic* if it contains no directed cycle. Note that in such a case every strong component consists of a single node.

[1] This terminology comes from the area of artificial intelligence, where the clause C is thought of as a "rule" $S \longrightarrow y$.

If $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ is a directed graph, then the *transitive closure* of $\mathbf{D}$ is a directed graph $\overline{\mathbf{D}} = (\mathbf{N}, \overline{\mathbf{A}})$ where $(x, y) \in \overline{\mathbf{A}}$, whenever there is a directed path from x to y in the digraph $\mathbf{D}$. Obviously, for a given $\mathbf{D}$ the transitive closure $\overline{\mathbf{D}}$ is uniquely defined. Finally, if $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ is a directed graph with strong components $C_1, \ldots, C_s$, then the directed graph $\mathbf{D}' = (\mathbf{N}', \mathbf{A}')$ on the set of nodes $\mathbf{N}' = \{C_1, \ldots, C_s\}$ with arcs

$$(C_i, C_j) \in \mathbf{A}' \quad \text{iff} \quad \exists x \in C_i \ \exists y \in C_j \text{ such that } (x, y) \in \mathbf{A}$$

is called the *acyclic condensation* of the digraph $\mathbf{D}$.

Definition 3.1. For a Horn CNF ϕ let $\mathbf{G}_\phi = (\mathbf{N}, \mathbf{A}_\phi)$ be the digraph defined by

$$\mathbf{N} = \{x \mid x \text{ is a propositional variable in } \phi\}$$
$$\mathbf{A}_\phi = \{(x, y) \mid \exists \text{ a clause } C \in \mathcal{C}(\phi) \text{ such that } C \geq \bar{x} \vee y\}.$$

In other words, for each pure Horn clause C in ϕ, the graph $\mathbf{G}_\phi$ contains as many arcs as the number of subgoals in C, with each arc going from the corresponding subgoal to the head of C. Since a Horn function can be represented by several different Horn CNFs, seemingly we can associate in this way several different graphs to a Horn function. However, as it was shown in [3], all these graphs share several important features.

Theorem 3.2. *Let ϕ_1 and ϕ_2 be two distinct prime CNFs representing the same Horn function f and let x, y be arbitrary propositional variables from f. Then there is a directed path from x to y in $\mathbf{G}_{\phi_1}$ if and only if there is a directed path from x to y in $\mathbf{G}_{\phi_2}$. Moreover, it then follows that $\mathbf{G}_{\phi_1}$ and $\mathbf{G}_{\phi_2}$ have identical transitive closures, identical strong components, and identical acyclic condensations.*

Theorem 3.2 allows us to associate a graph directly to a Horn function rather than to its particular CNF representations.

Definition 3.3. Let f be a Horn function and ϕ its arbitrary prime CNF representation. Then we define $\mathbf{G}_f$ as the transitive closure of $\mathbf{G}_\phi$.

The graph $\mathbf{G}_f$ moreover has the following useful property with respect to the set $\mathcal{I}(f)$.

Lemma 3.4. *Let $C = \bar{x}_1 \vee \bar{x}_2 \vee \ldots \vee \bar{x}_k \vee y \in \mathcal{I}(f)$ be arbitrary. Then (x_i, y) is an arc in $\mathbf{G}_f$ for every $1 \leq i \leq k$.*

Proof. If C is prime then it suffices to pick ϕ containing C in Definition 3.3 and the claim for C follows. Now observe that resolving two clauses in which each subgoal is connected to the head by an arc in $\mathbf{G}_f$ produces a resolvent in which each subgoal is connected to the head by a directed path in $\mathbf{G}_f$. However, $\mathbf{G}_f$ is transitively closed and thus the claim follows for any $C \in \mathcal{I}(f)$. □

Remark 3.5. Note that the correspondence between function f and graph G_f is not one-to-one. Consider e.g. f_1 represented by a single cubic clause $(\bar{a} \vee \bar{b} \vee c)$ and f_2 represented by two quadratic clauses $(\bar{a} \vee c) \wedge (\bar{b} \vee c)$ for which obviously $f_1 \neq f_2$ while $G_{f_1} = G_{f_2}$.

Theorem 3.2 suggests that for a given Horn function f the strong components of $\mathbf{G}_f$ play an important role in how the set of all prime CNF representations of f is structured. In what follows we shall call $\mathbf{G}_\phi$ and $\mathbf{G}_f$ the *implication graphs* of ϕ and f, respectively. The notion of implication graph of a Horn function f carries a lot of information about the CNF representations of f, allowing the characterization of two important subclasses of Horn functions.

A Horn CNF ϕ is said to be *acyclic* if its associated implication graph $\mathbf{G}_\phi$ is acyclic. A Horn function f is called *acyclic* if it admits at least one acyclic CNF representation. It was shown in [14] that every pure Horn acyclic function has a unique irredundant and prime representation. This fact has an important consequence: obviously, this unique CNF is also the shortest possible CNF representing the given pure Horn function. Moreover, as transforming an arbitrary Horn CNF into an equivalent prime and irredundant CNF can be done in polynomial time [10], this unique minimal CNF can be found efficiently. It was also shown in [10] that in fact this procedure can be used to minimize any acyclic Horn CNF (not necessarily pure Horn).

Let us call two propositional variables x and y *logically equivalent* in a Horn function f if the clauses $\bar{x} \vee y$ and $\bar{y} \vee x$ are implicates of f. A Horn CNF ϕ is then said to be *quasi-acyclic* (see [14]) if every strong component of its associated implication graph $\mathbf{G}_\phi$ consists of a set of logically equivalent propositional variables. A Horn function f is called *quasi-acyclic* if it admits at least one quasi-acyclic CNF representation.

Note that every acyclic CNF ϕ is quasi-acyclic since each strong component of $\mathbf{G}_\phi$ is a singleton. Also note that every *quadratic* Horn CNF ϕ (i.e. a CNF consisting of clauses of degree at most two) is quasi-acyclic as every strong component of $\mathbf{G}_\phi$ in this case consists of logically equivalent variables.

The name "quasi-acyclic" comes from the property that picking a representative in each set of logically equivalent propositional variables and substituting this representative for all the other logically equivalent variables in the set results in an acyclic CNF (i.e. the CNF is essentially acyclic except for the fact that each variable can have several "names"). In order to understand the structure of quasi-acyclic functions it is important to realize that if f is a quasi-acyclic function and x, y are propositional variables from the same strong component of $\mathbf{G}_f$ then both $\bar{x} \vee y$ and $\bar{y} \vee x$ are implicates of f. Hence no prime pure Horn implicate of f with degree three or more may contain a subgoal from the same strong component of $\mathbf{G}_f$ as the head. This means that the pure Horn clauses in any prime CNF representation of f can be partitioned into two groups. The first group (let us call it group A) contains clauses where all the subgoals are in different strong component(s) of $\mathbf{G}_f$ than the head, while the second group (group B) contains quadratic clauses with both the subgoal and the head belonging to the same strong component of $\mathbf{G}_f$. Loosely speaking, the clauses in group B "generate" the strong components of $\mathbf{G}_f$ while the clauses in group A "generate" its acyclic condensation. It was proved in [14] that similarly to the acyclic functions, a shortest CNF representation (i.e a representation with the minimum possible number of clauses) of a given quasi-acyclic function can be found in polynomial time.

Let us now leave the class of Horn functions and return (for the next three sections) to general Boolean functions.

4. Properties of resolution closures

In this section we prove several simple properties of resolution closures of sets of clauses and of CNF representations of Boolean functions. In the remainder of this section let us consider an arbitrary but fixed Boolean function f, the set $\mathcal{I}^p(f)$ of all prime implicates of f, and the set $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}^p(f))$ of all implicates of f that can be generated from the prime implicates of f by series of resolutions. Note that $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}(f))$, i.e. the set $\mathcal{I}(f)$ is closed under resolution.

Lemma 4.1. *If $\mathcal{A} \subseteq \mathcal{B}$ are arbitrary sets of clauses, then we have $\mathcal{R}(\mathcal{A}) \subseteq \mathcal{R}(\mathcal{B})$ and $\mathcal{R}(\mathcal{R}(\mathcal{A})) = \mathcal{R}(\mathcal{A})$.*

Proof. Immediate by the definition of the resolution closure. □

Lemma 4.2. *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two sets of clauses. Then $\mathcal{R}(\mathcal{C}_1) = \mathcal{R}(\mathcal{C}_2)$ implies that $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{C}_2)$, i.e. if the sets have the same resolution closure then they represent the same function.*

Proof. By Lemma 2.2, adding a resolvent of two clauses present in a given CNF to this CNF does not change the function being represented. Hence $\phi(\mathcal{C}) \equiv \phi(\mathcal{R}(\mathcal{C}))$ holds for any set $\mathcal{C}$ of clauses. Therefore $\mathcal{R}(\mathcal{C}_1) = \mathcal{R}(\mathcal{C}_2)$ implies $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{R}(\mathcal{C}_1)) = \phi(\mathcal{R}(\mathcal{C}_2)) \equiv \phi(\mathcal{C}_2)$ which completes the proof. □

Let us note that the reverse implication in the above statement is not valid. This is due to the fact that if we do not assume $\mathcal{C} \subseteq \mathcal{I}(f)$, then we can still have $\phi(\mathcal{C}) = f$ while $\mathcal{R}(\mathcal{C}) \neq \mathcal{I}(f)$. For instance, we have $(x \vee y) \wedge x \equiv x$, but the resolution closures of these two CNFs are not the same, obviously. On the other hand, adding the assumptions $\mathcal{C}_1 \subseteq \mathcal{I}(f)$ and $\mathcal{C}_2 \subseteq \mathcal{I}(f)$ makes the reverse implication work as well.

Lemma 4.3. *A set of clauses $\mathcal{C} \subseteq \mathcal{I}(f)$ represents the function f if and only if $\mathcal{I}(f) = \mathcal{R}(\mathcal{C})$.*

Proof. Assume first that $\mathcal{C} \subseteq \mathcal{I}(f)$ represents f. Then by Theorem 2.3 we have $\mathcal{I}^p(f) \subseteq \mathcal{R}(\mathcal{C})$, and thus $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}^p(f)) \subseteq \mathcal{R}(\mathcal{R}(\mathcal{C})) = \mathcal{R}(\mathcal{C}) \subseteq \mathcal{R}(\mathcal{I}(f)) = \mathcal{I}(f)$ follows by Lemma 4.1.

For the reverse direction assume that $\mathcal{I}(f) = \mathcal{R}(\mathcal{C})$. Now $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}(f))$ together with Lemma 4.2 imply $\phi(\mathcal{C}) \equiv \phi(\mathcal{I}(f))$ and so $\mathcal{C}$ represents f. □

The following two simple technical lemmas will be useful in our proofs later.

Lemma 4.4. *Let $\mathcal{V}, \mathcal{W} \subseteq \mathcal{I}(f)$ be two arbitrary sets of clauses. Then $\mathcal{R}(\mathcal{V} \cup \mathcal{W}) = \mathcal{R}(\mathcal{V} \cup \mathcal{R}(\mathcal{W}))$.*

Proof. The inclusion $\mathcal{R}(\mathcal{V} \cup \mathcal{W}) \subseteq \mathcal{R}(\mathcal{V} \cup \mathcal{R}(\mathcal{W}))$ follows by Lemma 4.1, since $\mathcal{V} \cup \mathcal{W} \subseteq \mathcal{V} \cup \mathcal{R}(\mathcal{W})$ by the definition of the resolution closure. To prove the opposite inclusion, let us assume that $C \in \mathcal{R}(\mathcal{V} \cup \mathcal{R}(\mathcal{W}))$. Then, any $C' \in \mathcal{R}(\mathcal{W})$ used in the derivation of C from the set $\mathcal{V} \cup \mathcal{R}(\mathcal{W})$ by a series of resolutions is, by assumption, itself derivable from the set $\mathcal{W}$, and hence $C \in \mathcal{R}(\mathcal{V} \cup \mathcal{W})$ holds. □

Let us close out this section with a lemma which roughly says the following: replacing a set of clauses $\mathcal{E}$ by its subset $\mathcal{Q}$ does not change the function if the resolution closures of $\mathcal{E}$ and $\mathcal{Q}$ are the same.

Lemma 4.5. *Let $\mathcal{C} \subseteq \mathcal{I}(f)$ and $\mathcal{Q} \subseteq \mathcal{E} \subseteq \mathcal{R}(\mathcal{C})$ be sets of clauses such that $\mathcal{R}(\mathcal{Q}) = \mathcal{R}(\mathcal{E})$. Then $\mathcal{R}((\mathcal{C} \setminus \mathcal{E}) \cup \mathcal{Q}) = \mathcal{R}(\mathcal{C})$.*

Proof. Using Lemma 4.4 with $\mathcal{V} = \mathcal{C} \setminus \mathcal{E}$ twice (first with $\mathcal{W} = \mathcal{Q}$ and then with $\mathcal{W} = \mathcal{E}$) we get

$$\mathcal{R}((\mathcal{C} \setminus \mathcal{E}) \cup \mathcal{Q}) = \mathcal{R}((\mathcal{C} \setminus \mathcal{E}) \cup \mathcal{R}(\mathcal{Q})) = \mathcal{R}((\mathcal{C} \setminus \mathcal{E}) \cup \mathcal{R}(\mathcal{E})) = \mathcal{R}((\mathcal{C} \setminus \mathcal{E}) \cup \mathcal{E}) = \mathcal{R}(\mathcal{C})$$

which is the stated result. □

5. Exclusive sets and exclusive components of functions

As in the previous section let us consider throughout this section an arbitrary but fixed Boolean function f and the sets of clauses $\mathcal{I}^p(f)$ and $\mathcal{I}(f)$ associated with it. Let us now define the first of the two key concepts of this paper.

Definition 5.1. Given a set $\mathcal{C}$ of clauses, a subset $\mathcal{X} \subseteq \mathcal{C}$ is called an *exclusive subset of* $\mathcal{C}$ if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{C}$ the following implication holds:

$$R(C_1, C_2) \in \mathcal{X} \Longrightarrow C_1 \in \mathcal{X} \quad \text{and} \quad C_2 \in \mathcal{X},$$

i.e. the resolvent belongs to $\mathcal{X}$ only if both parent clauses are in $\mathcal{X}$. In particular, if $\mathcal{C} = \mathcal{I}(f)$ for a Boolean function f, we call such a subset $\mathcal{X}$ an exclusive set of clauses of f (or simply an exclusive set, if f or $\mathcal{C}$ is clear from the context).

Note that the above definition trivially implies that $\mathcal{C}$ itself is the largest (with respect to inclusion) exclusive subset of $\mathcal{C}$ (for any $\mathcal{C}$), and in particular $\mathcal{I}(f)$ is by definition the largest exclusive set of clauses of f. Let us first claim in the next lemma some simple properties possessed by exclusive sets. Since all these properties follow directly from Definition 5.1 we shall omit the proofs.

Lemma 5.2. *Let $\mathcal{C}$ be an arbitrary set of clauses. Then,*

(a) *if $\mathcal{A}$ is an exclusive subset of $\mathcal{B}$ and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{A}$ is an exclusive subset of $\mathcal{C}$;*
(b) *if $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{C}$, and $\mathcal{A}$ is an exclusive subset of $\mathcal{C}$, then it is also an exclusive subset of $\mathcal{B}$;*
(c) *if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{C}$ are both exclusive subsets of $\mathcal{C}$, then $\mathcal{A} \cup \mathcal{B}$ and $\mathcal{A} \cap \mathcal{B}$ are also exclusive (and hence all exclusive subsets of $\mathcal{C}$ form a lattice).* $\square$

To see an interesting example of exclusive sets of clauses, let us for a moment return to Horn functions. Let h be a Horn function and let us partition the set $\mathcal{I}(h)$ into two subsets $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ where $\mathcal{H}$ is the set of all pure Horn clauses in $\mathcal{I}(h)$ and $\mathcal{N}$ is the set of all negative clauses in $\mathcal{I}(h)$.[2] Then it is not hard to see that $\mathcal{H}$ is an exclusive set of h (the resolvent is in $\mathcal{H}$ only if both parent clauses are in $\mathcal{H}$). Moreover, in this case any resolvent of two clauses in $\mathcal{H}$ is again in $\mathcal{H}$ which is not necessarily true for every exclusive set.

The partition $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ has some important properties, shown in [10]. The first such property states that if ϕ_1 and ϕ_2 are two distinct prime CNFs representing h, then the pure Horn parts of ϕ_1 and ϕ_2 (i.e. the conjunctions of all pure Horn clauses in the given CNFs) also represent the same pure Horn function, called in [10] the *pure Horn component* of h.

Proposition 5.3 ([10]). *Let ϕ_1 and ϕ_2 be two distinct prime CNFs of a Horn function h. Then $\phi(\mathcal{C}(\phi_1) \cap \mathcal{H}) \equiv \phi(\mathcal{C}(\phi_2) \cap \mathcal{H})$.*

The proof of this proposition is based on the above mentioned fact that the only way how to generate a pure Horn clause by resolution is to use two pure Horn clauses as the parent clauses. Using the just defined terminology, the proof rests on the fact that pure Horn clauses constitute an exclusive set of h. Thus it is obviously very tempting to generalize the result to all exclusive sets. However, first we need to state a simple lemma.

Lemma 5.4. *Let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses (of f) and $\mathcal{C} \subseteq \mathcal{I}(f)$ be a set of clauses such that $\mathcal{X} \subseteq \mathcal{R}(\mathcal{C})$. Then $\mathcal{R}(\mathcal{X}) = \mathcal{R}(\mathcal{C} \cap \mathcal{X})$.*

Proof. The fact that $\mathcal{X}$ is exclusive means that no clause in $\mathcal{C} \setminus \mathcal{X}$ can appear as a parent clause in a resolution leading to a resolvent in $\mathcal{X}$. That, together with the inclusion $\mathcal{X} \subseteq \mathcal{R}(\mathcal{C})$, implies that $\mathcal{X} \subseteq \mathcal{R}(\mathcal{C} \setminus (\mathcal{C} \setminus \mathcal{X})) = \mathcal{R}(\mathcal{C} \cap \mathcal{X})$ must hold, which in turn implies $\mathcal{R}(\mathcal{X}) \subseteq \mathcal{R}(\mathcal{R}(\mathcal{C} \cap \mathcal{X})) = \mathcal{R}(\mathcal{C} \cap \mathcal{X})$ by Lemma 4.1. The reverse inclusion $\mathcal{R}(\mathcal{C} \cap \mathcal{X}) \subseteq \mathcal{R}(\mathcal{X})$ follows also from Lemma 4.1, since $\mathcal{C} \cap \mathcal{X} \subseteq \mathcal{X}$ holds trivially. $\square$

We are now ready to generalize Proposition 5.3 to all exclusive sets.

Theorem 5.5. *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{I}(f)$ be two distinct sets of clauses such that $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{C}_2) \equiv f$, i.e. such that both sets represent f, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Then $\phi(\mathcal{C}_1 \cap \mathcal{X}) \equiv \phi(\mathcal{C}_2 \cap \mathcal{X})$.*

Proof. Since both $\mathcal{C}_1$ and $\mathcal{C}_2$ represent f, we have $\mathcal{R}(\mathcal{C}_1) = \mathcal{R}(\mathcal{C}_2) = \mathcal{I}(f)$ by Lemma 4.3, and hence $\mathcal{X} \subseteq \mathcal{R}(\mathcal{C}_1) = \mathcal{R}(\mathcal{C}_2)$. Now by Lemma 5.4 we get $\mathcal{R}(\mathcal{X}) = \mathcal{R}(\mathcal{C}_1 \cap \mathcal{X}) = \mathcal{R}(\mathcal{C}_2 \cap \mathcal{X})$ which then implies $\phi(\mathcal{C}_1 \cap \mathcal{X}) \equiv \phi(\mathcal{C}_2 \cap \mathcal{X})$ by Lemma 4.2. $\square$

It is immediate to see that Proposition 5.3 is just a special case of Theorem 5.5. We can also generalize the notion of a "pure Horn component".

Definition 5.6. Let f be an arbitrary Boolean function, $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses of f, and $\mathcal{C} \subseteq \mathcal{I}(f)$ be a set of clauses which represents f (i.e. $\phi(\mathcal{C}) \equiv f$). The Boolean function $f_{\mathcal{X}}$ represented by the set $\mathcal{C} \cap \mathcal{X}$ is called the $\mathcal{X}$-*component* of the function f. We shall simply call a function g an *exclusive component of f*, if $g = f_{\mathcal{X}}$ for some exclusive subset $\mathcal{X} \subseteq \mathcal{I}(f)$.

Theorem 5.5 guarantees that the $\mathcal{X}$-component $f_{\mathcal{X}}$ is well defined for every exclusive set $\mathcal{X} \subseteq \mathcal{I}(f)$. Let us now briefly return to Proposition 5.3. It has the following consequence: given a Horn CNF one can extract the pure Horn sub-CNF which represents the pure Horn component, find its shortest CNF representation, and then insert this new sub-CNF back into the input CNF. This is exactly how the minimization procedure for acyclic and quasi-acyclic functions works. A similar (but more general) consequence can be drawn from Theorem 5.5.

Corollary 5.7. *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{I}(f)$ be two distinct sets of clauses such that $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{C}_2) \equiv f$, i.e. such that both sets represent f, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Then $\phi((\mathcal{C}_1 \setminus \mathcal{X}) \cup (\mathcal{C}_2 \cap \mathcal{X})) \equiv f$.*

Proof. Similarly as in the proof of Theorem 5.5 we get $\mathcal{X} \subseteq \mathcal{R}(\mathcal{C}_1) = \mathcal{R}(\mathcal{C}_2)$ and also $\mathcal{R}(\mathcal{X}) = \mathcal{R}(\mathcal{C}_1 \cap \mathcal{X}) = \mathcal{R}(\mathcal{C}_2 \cap \mathcal{X})$. Now using Lemma 4.5 with $\mathcal{Q} = \mathcal{C}_2 \cap \mathcal{X}$, $\mathcal{E} = \mathcal{X}$, and $\mathcal{C} = \mathcal{C}_1$ (note that $\mathcal{C}_2 \cap \mathcal{X} \subseteq \mathcal{X} \subseteq \mathcal{R}(\mathcal{C}_1)$ and so the assumptions of Lemma 4.5 are satisfied) we obtain

$$\mathcal{R}((\mathcal{C}_1 \setminus \mathcal{X}) \cup (\mathcal{C}_2 \cap \mathcal{X})) = \mathcal{R}(\mathcal{C}_1)$$

which together with the fact $\mathcal{R}(\mathcal{C}_1) = \mathcal{R}(\mathcal{C}_2) = \mathcal{I}(f)$ completes the proof by Lemma 4.2. $\square$

[2] It is left to the reader to verify the easy fact that $\mathcal{H}$ and $\mathcal{N}$ indeed constitute a partition of $\mathcal{I}(h)$, i.e. that no clause which is neither pure Horn nor negative can appear in $\mathcal{I}(h)$ (recall that each prime implicate of a Horn function is either pure Horn or negative).

The algorithmic meaning of Corollary 5.7 can be stated as follows. If $\mathcal{C}_1$ is the input CNF, one can extract the sub-CNF $\mathcal{C}_1 \cap \mathcal{X}$ which represents the $\mathcal{X}$-component $f_{\mathcal{X}}$, find its shortest CNF representation (say $\mathcal{C}_2 \cap \mathcal{X}$), and then insert this new sub-CNF back into the input CNF. That suggests a decomposition approach for minimization algorithms. Whenever one can find an exclusive subset of clauses of a given function or several pairwise disjoint exclusive subsets of clauses of a given function, it is possible to decompose the minimization problem, solve the subproblems separately, and then compose the obtained solutions back together.

Let us close this section by a simple corollary about certain redundant sets of clauses. A clause $C \in \mathcal{I}(f)$ is called *redundant* with respect to a function f if C does not appear in any irredundant CNF representation $\mathcal{C} \subseteq \mathcal{I}(f)$ of f (in other words, $C \notin \mathcal{C}$ for any minimal $\mathcal{C} \subseteq \mathcal{I}(f)$ such that $\mathcal{R}(\mathcal{C}) = \mathcal{I}(f)$). A set $\mathcal{S} \subseteq \mathcal{I}(f)$ of clauses is called *redundant* with respect to a function f if every clause in $\mathcal{S}$ is redundant with respect to f, i.e., if $\mathcal{S} \cap \mathcal{C} = \emptyset$ for every irredundant representation $\mathcal{C} \subseteq \mathcal{I}(f)$ of f.

Corollary 5.8. *For every exclusive set $\mathcal{X} \subseteq \mathcal{I}(f)$ we have $\mathcal{R}(\mathcal{X}) = \mathcal{I}(f_{\mathcal{X}})$, furthermore the set $\mathcal{R}(\mathcal{X}) \setminus \mathcal{X}$ is redundant with respect to $f_{\mathcal{X}}$, as well as with respect to f.*

Proof. By Lemma 5.4 we have $\mathcal{R}(\mathcal{X}) = \mathcal{R}(\mathcal{X} \cap \mathcal{C})$ for every irredundant representation $\mathcal{C} \subseteq \mathcal{I}(f)$ of f, and hence $\mathcal{C} \cap (\mathcal{R}(\mathcal{X}) \setminus \mathcal{X}) = \emptyset$ follows for all such representations by their irredundancy. Furthermore, by Corollary 5.7 and Lemma 4.5, we can choose $\mathcal{C}$ such that $\mathcal{C} \cap \mathcal{X}$ is a prime irredundant representation of $f_{\mathcal{X}}$, from which $\mathcal{I}(f_{\mathcal{X}}) = \mathcal{R}(\mathcal{X} \cap \mathcal{C}) = \mathcal{R}(\mathcal{X})$ follows (again by Lemma 5.4). □

6. Essential sets and a min–max relation

Let us now introduce the second key concept of this paper.

Definition 6.1. Given a set $\mathcal{C}$ of clauses, a subset $\mathcal{E} \subseteq \mathcal{C}$ is called an *essential subset of* $\mathcal{C}$ if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{C}$ the following implication holds:

$$R(C_1, C_2) \in \mathcal{E} \implies C_1 \in \mathcal{E} \text{ or } C_2 \in \mathcal{E},$$

i.e. the resolvent belongs to $\mathcal{E}$ only if at least one of the parent clauses is from $\mathcal{E}$. In particular, if $\mathcal{C} = \mathcal{I}(f)$ for a Boolean function f, we call $\mathcal{E}$ an essential set of clauses of f (or simply an essential set, if f or $\mathcal{C}$ is clear from the context).

It is easy to see that every exclusive set of clauses (and the set $\mathcal{I}(f)$ in particular) is also essential. We summarize in the following lemma a few simple properties of essential sets. Since all these properties follow directly from Definitions 5.1 and 6.1 we shall omit the proofs.

Lemma 6.2. *Let $\mathcal{C}$ be an arbitrary set of clauses. Then,*

(a) *if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{C}$ are both essential subsets of $\mathcal{C}$, then $\mathcal{A} \cup \mathcal{B}$ is also essential;*
(b) *if $\mathcal{R}(\mathcal{C}) = \mathcal{C}$ and $\mathcal{A}$ is an essential subset of $\mathcal{C}$, then $\mathcal{C} \setminus \mathcal{A}$ is closed under resolution, i.e. $\mathcal{C} \setminus \mathcal{A} = \mathcal{R}(\mathcal{C} \setminus \mathcal{A})$;*
(c) *if $\mathcal{R}(\mathcal{A}) = \mathcal{A}$ and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{B} \setminus \mathcal{A}$ is an essential subset of $\mathcal{C}$;*
(d) *if $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{C}$, $\mathcal{A}$ is an essential subset of $\mathcal{B}$, and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{A}$ is an essential subset of $\mathcal{C}$, as well;*
(e) *if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{C}$, $\mathcal{A} \cap \mathcal{B} \neq \emptyset$, $\mathcal{A}$ is an essential subset of $\mathcal{C}$, and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{A} \cap \mathcal{B}$ is also an essential subset of $\mathcal{C}$.* □

To see an interesting example of essential sets, let us consider again a Horn function h and return to the partition of the set $\mathcal{I}(h)$ into two subsets $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ where $\mathcal{H}$ is the set of all pure Horn clauses in $\mathcal{I}(h)$ and $\mathcal{N}$ is the set of all negative clauses in $\mathcal{I}(h)$. Then, it is not hard to see that $\mathcal{N}$ is essential for h. In fact, more is true in this case: since no two clauses in $\mathcal{N}$ are resolvable, the resolvent is in $\mathcal{N}$ only if *exactly* one of the parent clauses is in $\mathcal{N}$ and the other one is in $\mathcal{H}$.

A second important property (aside of Proposition 5.3) of the partition $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ states that if ϕ_1 and ϕ_2 are two distinct irredundant CNFs representing h, then ϕ_1 and ϕ_2 both contain the same number of negative clauses.

Proposition 6.3 ([10]). *Let ϕ_1 and ϕ_2 be two distinct irredundant CNFs of a Horn function h. Then $|\mathcal{C}(\phi_1) \cap \mathcal{N}| = |\mathcal{C}(\phi_2) \cap \mathcal{N}|$.*

The proof of this proposition is more or less based on the above mentioned fact that negative clauses form an essential set of h with no resolvable pair, and their complement (i.e. pure Horn clauses) forms an exclusive set of h (of course the original proof did not use this terminology). This observation leads to an obvious idea to generalize the statement of Proposition 6.3. We will arrive at such a generalization in the end of this section (Theorem 6.16 and Corollary 6.17).

Another way to look at the above example of an essential set is to realize that it is a special case of Lemma 6.2 case (c) where we take $\mathcal{A} = \mathcal{H}$ and $\mathcal{B} = \mathcal{C} = \mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ (note that $\mathcal{R}(\mathcal{H}) = \mathcal{H}$). A similar statement is of course true for any function f: if for some $\mathcal{A} \subseteq \mathcal{I}(f)$ we have $\mathcal{R}(\mathcal{A}) = \mathcal{A}$ then $\mathcal{I}(f) \setminus \mathcal{A}$ is an essential set. Note that f is not required to be Horn, so this gives us examples of non-Horn essential sets (some of them will be discussed in more detail in Theorem 6.7).

Another example of essential sets is given by the following definition.

Definition 6.4. Given a Boolean function f on n propositional variables and an arbitrary vector $t \in \{0, 1\}^n$ let us define a *falsepoint set of f defined by t* as

$$\mathcal{E}(t) = \{C \in \mathcal{I}(f) \mid C(t) = 0\}$$

where by $C(t) = 0$ we mean the following: if we substitute for the propositional variables of f the truth values according to the vector t then the clause C evaluates to zero (false).

Lemma 6.5. *Let f be a Boolean function on n propositional variables and let $t \in \{0, 1\}^n$ be an arbitrary vector. Then $\mathcal{E}(t)$ is an essential set of f.*

Proof. By definition, $\mathcal{E}(t) \subseteq \mathcal{I}(f)$. Let $C_1 = A \vee x$ and $C_2 = B \vee \bar{x}$ be arbitrary two resolvable clauses from $\mathcal{I}(f)$ such that $R(C_1, C_2) = A \vee B \in \mathcal{E}(t)$. This means that all literals in the set $A \cup B$ evaluate to zero under the truth assignment t, which in turn implies that exactly one of the clauses C_1 and C_2 evaluates to zero, depending on the value assigned to x. Therefore either $C_1 \in \mathcal{E}(t)$ or $C_2 \in \mathcal{E}(t)$ and thus $\mathcal{E}(t)$ is essential. □

We shall use Lemma 6.5 to prove a key theorem which shows that every essential set has one (or more) of its clauses present in every representation of f and moreover that this condition is not only necessary but also sufficient.

Theorem 6.6. *Let $\mathcal{C} \subseteq \mathcal{I}(f)$ be an arbitrary set of clauses. Then $\mathcal{C}$ represents f if and only if $\mathcal{C} \cap \mathcal{E} \neq \emptyset$ for every nonempty essential set of clauses $\mathcal{E} \subseteq \mathcal{I}(f)$.*

Proof. Let us assume that $\mathcal{C} \subseteq \mathcal{I}(f)$ represents f and $\mathcal{C} \cap \mathcal{E} = \emptyset$ for some nonempty essential set $\mathcal{E} \subseteq \mathcal{I}(f)$. That means that when we start making resolutions from the set $\mathcal{C}$ we can never get into $\mathcal{E}$, i.e. that $\mathcal{R}(\mathcal{C}) \cap \mathcal{E} = \emptyset$. However, $\mathcal{E} \subseteq \mathcal{I}(f) = \mathcal{R}(\mathcal{C})$ by Lemma 4.3, which is a contradiction. Therefore $\mathcal{C} \cap \mathcal{E} \neq \emptyset$ for every nonempty essential set of clauses $\mathcal{E} \subseteq \mathcal{I}(f)$.

To prove the opposite implication let us assume that $\mathcal{C} \subseteq \mathcal{I}(f)$ is a set of clauses which has a nonempty intersection with every nonempty essential set $\mathcal{E} \subseteq \mathcal{I}(f)$. By Lemma 6.5 $\mathcal{C} \cap \mathcal{E}(t) \neq \emptyset$ holds for every truth assignment t such that $\mathcal{E}(t)$ is nonempty. Let us now denote the CNF $\phi(\mathcal{C})$ by ϕ. We want to prove $\phi \equiv f$. The inequality $\phi \geq f$ trivially follows from the fact that $\mathcal{C} \subseteq \mathcal{I}(f)$. Thus it remains to be shown that also $\phi \leq f$ which is equivalent to proving that $f(t) = 0$ implies $\phi(t) = 0$ for every truth assignment t. So let t be an arbitrary assignment such that $f(t) = 0$. This means that there must exist a prime implicate C of f such that $C(t) = 0$. Therefore $\mathcal{E}(t)$ is a nonempty set and by Lemma 6.5 also an essential set, and thus by our assumption $\mathcal{C} \cap \mathcal{E}(t) \neq \emptyset$ holds, i.e. $\mathcal{C}$ contains a clause that evaluates to zero under the truth assignment t. However, that implies $\phi(t) = 0$ which completes the proof. □

Some of the sets $\mathcal{E}(t)$ defined in Definition 6.4 play quite an important role in the structure of $\mathcal{I}(f)$. Let us for a moment consider the lattice $\mathcal{L}$ of all subsets of $\mathcal{I}(f)$. Clearly, the property of being a representation of f is monotone in $\mathcal{L}$ (every superset of a representation is again a representation). The minimal sets in $\mathcal{L}$ which represent f are of course exactly all the irredundant representations of f. On the other hand, also the property of not being a representation of f is a monotone one in $\mathcal{L}$ (every subset of a non-representation is again a non-representation). Thus it is of some interest to understand what the maximal non-representations of f in $\mathcal{L}$ are. We summarize the properties of these sets in the following theorem.

Theorem 6.7. *Let $\mathcal{D} \subseteq \mathcal{I}(f)$ be a maximal (under inclusion) set of clauses not representing f. Then $\mathcal{D} = \mathcal{R}(\mathcal{D})$, the set $\mathcal{I}(f) \setminus \mathcal{D}$ is essential, and there exists a Boolean vector t such that $\mathcal{I}(f) \setminus \mathcal{D} = \mathcal{E}(t)$.*

Proof. Let us assume that there exist clauses $C_1, C_2 \in \mathcal{D}$ such that $C = R(C_1, C_2) \notin \mathcal{D}$. By the property of resolution (Lemma 2.2) we have $\phi(\mathcal{D}) = \phi(\mathcal{D} \cup \{C\})$, and hence $\mathcal{D} \cup \{C\}$ still does not represent f, which is a contradiction to the maximality of $\mathcal{D}$. Therefore $\mathcal{D} = \mathcal{R}(\mathcal{D})$ and by Lemma 6.2 part (c) (where we take $\mathcal{A} = \mathcal{D}$ and $\mathcal{B} = \mathcal{C} = \mathcal{I}(f)$) the set $\mathcal{I}(f) \setminus \mathcal{D}$ is essential.

To finish the proof let us consider the function $f' = \phi(\mathcal{D})$. Clearly $f \leq f'$ and since $f \neq f'$ there must exist a Boolean vector t such that $f(t) = 0$ and $f'(t) = 1$. Now consider the set $\mathcal{E}(t)$. Obviously $\mathcal{E}(t) \cap \mathcal{D} = \emptyset$ (all clauses in $\mathcal{D}$ must evaluate to 1 on t in order to make $f'(t) = 1$). So it remains to prove that $\mathcal{E}(t) \cup \mathcal{D} = \mathcal{I}(f)$, or in other words that every clause that evaluates to 1 on t is in $\mathcal{D}$. Assume by contradiction that there exists a clause $C \notin \mathcal{D}$ such that $C \in \mathcal{I}(f)$ and $C(t) = 1$. Clearly $\phi(\mathcal{D} \cup \{C\})(t) = 1$ while $f(t) = 0$ and so the set $\mathcal{D} \cup \{C\}$ still does not represent f, which is again a contradiction to the maximality of $\mathcal{D}$. □

It follows from the proof of Theorem 6.7 that the maximal non-representations of f in $\mathcal{L}$ can be alternatively characterized as follows.

Corollary 6.8. *Set $\mathcal{D} \subseteq \mathcal{I}(f)$ of clauses is a maximal (under inclusion) set not representing f if and only if $\mathcal{D}$ is a maximal proper subset of $\mathcal{I}(f)$ closed under resolution.*

To give one more example of essential sets let us recall the following definition.

Definition 6.9. Given a Boolean function f a clause C is called an *essential prime implicate of f* if C appears in every prime and irredundant CNF representation of f.

We shall show that essential prime implicates are just a special case of essential sets of cardinality one.

Lemma 6.10. *Let f be a Boolean function and C its prime implicate. Then C is an essential prime implicate of f if and only if $\{C\}$ is an essential set of f.*

Proof. If $\{C\}$ is an essential set of f, then by Theorem 6.6 clause C must be present in every CNF φ representing f, i.e. C is an essential prime implicate of f.

On the other hand let C be an essential prime implicate of f, it follows, that $\mathcal{I}^p(f) \setminus \{C\}$ does not represent f and hence there is a vector t, for which $C(t) = 0$ while $C'(t) = 1$ for every $C' \in \mathcal{I}^p(f) \setminus \{C\}$. Let us examine set $\mathcal{E}(t)$, which is an essential set due to Lemma 6.5. We shall show that $\mathcal{E}(t) = \{C\}$. Since C is the only prime implicate which belongs to $\mathcal{E}(t)$, we can observe, that in fact

$$\mathcal{E}(t) \subseteq \{D \in \mathcal{I}(f) \mid C \leq D\}$$

i.e. all clauses in $\mathcal{E}(t)$ contain C. Let us proceed by contradiction and let us assume $|\mathcal{E}(t)| > 1$. Every clause in $\mathcal{E}(t) \subseteq \mathcal{I}(f)$ can be derived from $\mathcal{I}^p(f)$ by a series of resolutions. Let us denote by D the clause from $\mathcal{E}(t) \setminus \{C\}$ for which such a derivation is the shortest possible. Moreover, let $D = R(X, Y)$ be the last step of such a derivation. Since $\mathcal{E}(t)$ is essential, one of the parent clauses must be in $\mathcal{E}(t)$, w.l.o.g. let $X \in \mathcal{E}(t)$. Now if $X \neq C$, then $X \in \mathcal{E}(t) \setminus \{C\}$ and its derivation from $\mathcal{I}^p(f)$ is shorter than the derivation of D contradicting the choice of D. Hence $X = C$. But now $D = R(C, Y)$ contradicts $C \leq D$. Therefore no such D can exist and we have shown, that $\mathcal{E}(t) = \{C\}$. $\square$

Let us now return to Theorem 6.6. It has an obvious corollary: if there exist nonempty essential sets $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_k \subseteq \mathcal{I}(f)$ which are pairwise disjoint, then every representation of f must consist of at least k clauses. Hence, any collection of pairwise disjoint essential sets of clauses provides an easy lower bound on the size (i.e. on the number of clauses) of a minimal representation of f.

Definition 6.11. Given a Boolean function f, let us denote by $\epsilon(f)$ the maximum number of pairwise disjoint nonempty essential subsets of $\mathcal{I}(f)$ and by $\sigma(f)$ the minimum number of clauses needed to represent f by a CNF.

Using this notation, we can now formulate the above noted simple corollary of Theorem 6.6 very succinctly as follows:

Corollary 6.12. *For every Boolean function f the following inequality holds*

$$\sigma(f) \geq \epsilon(f).$$

Note that Corollary 6.12 generalizes the known lower bound on the size of a CNF representation given by the number of essential prime implicates. Indeed, by Lemma 6.10 the one element sets defined by essential prime implicates form a collection of pairwise disjoint essential sets, and hence $\epsilon(f)$ is always greater or equal to the number of essential prime implicates.

It remains to be seen for which classes of functions the inequality in Corollary 6.12 is tight, i.e. turns into an equality. Let us state this formally as a question.

Question 6.13. *Let $\mathcal{C}$ be a class of Boolean functions. Is it true that $\sigma(f) = \epsilon(f)$ holds for every $f \in \mathcal{C}$?*

In the next section we shall prove that Question 6.13 has an affirmative answer for the following three subclasses of Horn functions (which were introduced in Section 3): quadratic Horn functions, acyclic Horn functions, and quasi-acyclic Horn functions.[3]

Let us now come back to Theorem 6.6. The first part of the proof of Theorem 6.6, which shows that for an arbitrary $\mathcal{C}$ representing f and an arbitrary nonempty essential set $\mathcal{E}$, $\mathcal{C} \cap \mathcal{E} \neq \emptyset$ must hold, gives yet another corollary.

Corollary 6.14. *Let $\mathcal{E} \subseteq \mathcal{I}(f)$ be an arbitrary set of clauses. Then $\mathcal{E}$ is a nonempty essential set only if $\mathcal{E} \cap \mathcal{C} \neq \emptyset$ for every representation $\mathcal{C} \subseteq \mathcal{I}(f)$ of the function f.*

There is an obvious duality between Corollary 6.14 and Theorem 6.6 based on the reversal of roles between $\mathcal{E}$ and $\mathcal{C}$. However, unlike in Theorem 6.6 where the equivalence holds, only one implication (the "only if" part) is true in Corollary 6.14. The reason for this is the following. If $\mathcal{C}$ represents f and we add some clauses from $\mathcal{I}(f)$ to $\mathcal{C}$, the new set will still represent f. On the other hand, if $\mathcal{E}$ is an essential set and we add some clauses from $\mathcal{I}(f)$ to $\mathcal{E}$, the resulting set may not be essential. To avoid the presence of such "extra" clauses in $\mathcal{E}$ we shall add a minimality assumption. This minimality assumption simply means that we shall require not only that $\mathcal{E}$ intersect all representations of f (i.e. that $\mathcal{E}$ be a transversal of all representations) but also that $\mathcal{E}$ be a minimal set with this property (i.e. that $\mathcal{E}$ be a minimal transversal). With this additional assumption the reverse implication in Corollary 6.14 (the "if" part) becomes valid as well, making the duality with Theorem 6.6 work both ways (i.e., informally speaking, by the above corollary every essential set forms a transversal of all representations, and by the theorem below every minimal transversal of all representations forms an essential set).

Theorem 6.15. *Let $\mathcal{E}$ be an arbitrary minimal (under inclusion) subset of $\mathcal{I}(f)$ such that $\mathcal{E} \cap \mathcal{C} \neq \emptyset$ for every $\mathcal{C} \subseteq \mathcal{I}(f)$ which represents f. Then $\mathcal{E}$ is an essential set of clauses.*

[3] There exists a counterexample proving that the equality does not hold for the class of all Horn functions [19].

Proof. Let us assume by contradiction that $\mathcal{E}$ is not essential, i.e. that there exist two resolvable clauses $C_1, C_2 \in \mathcal{I}(f)$ such that $C_1, C_2 \notin \mathcal{E}$ but $C = R(C_1, C_2) \in \mathcal{E}$. Let us consider the set Σ of all representations of f which contain the clause C, i.e. let

$$\Sigma = \{\mathcal{C} \mid \mathcal{R}(\mathcal{C}) = \mathcal{I}(f) \text{ and } C \in \mathcal{C}\}.$$

If every $\mathcal{C} \in \Sigma$ intersected $\mathcal{E}$ in two or more clauses then we could remove C from $\mathcal{E}$ and still maintain the property that $\mathcal{E}$ intersects all representations of f. However, this would be a contradiction to the minimality of $\mathcal{E}$. Therefore there must exist a representation $\mathcal{C}'$ of f in the set Σ which intersects $\mathcal{E}$ in exactly one clause, i.e. such that $\mathcal{C}' \cap \mathcal{E} = \{C\}$. Let us now construct a set of clauses

$$\mathcal{C}'' = \mathcal{C}' \setminus \{C\} \cup \{C_1, C_2\}.$$

Clearly $\mathcal{C}'' \subseteq \mathcal{I}(f)$. Moreover, since $C = R(C_1, C_2)$ it is obvious that $\mathcal{R}(\mathcal{C}'') = \mathcal{R}(\mathcal{C}) = \mathcal{I}(f)$, i.e., $\mathcal{C}''$ represents f. However, $\mathcal{C}'' \cap \mathcal{E} = \emptyset$ which is a contradiction to the choice of $\mathcal{E}$. □

Let us finish this section by proving a generalization of Proposition 6.3.

Theorem 6.16. *Given a Boolean function f, let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive subset of f such that no two clauses from $\mathcal{E} = \mathcal{I}(f) \setminus \mathcal{R}(\mathcal{X})$ are resolvable. Then, there exists an integer $k = k(\mathcal{E}) > 0$, and pairwise disjoint essential subsets $\mathcal{Q}_j \subseteq \mathcal{E}$, $j = 1, .., k$ such that $|\mathcal{Q}_j \cap \mathcal{C}| = 1$ for $j = 1, \dots, k$ and $|(\mathcal{E} \setminus \bigcup_{j=1}^{k} \mathcal{Q}_j) \cap \mathcal{C}| = 0$ for any irredundant set $\mathcal{C} \subseteq \mathcal{I}(f)$ of clauses representing f.*

Proof. Let us observe that $\mathcal{E}$ is an essential set by (c) of Lemma 6.2. Furthermore, the property that no two clauses of the essential family $\mathcal{E}$ are resolvable implies that if $R(A, B) \in \mathcal{E}$ for some resolvable clauses $A, B \in \mathcal{I}(f)$, then exactly one of these clauses belongs to $\mathcal{E}$.

Let us then first define a directed graph $\mathbf{H}$, the vertices of which are the clauses in $\mathcal{E}$, and where (A, B) is a directed arc for $A, B \in \mathcal{E}$ if and only if $B \in \mathcal{R}(\mathcal{X} \cup \{A\})$. Let us next consider a strong component $\mathcal{Q} \subseteq \mathcal{E}$ of $\mathbf{H}$, which is an initial component, i.e., for which there exists no arc (A, B) of $\mathbf{H}$ such that $A \in \mathcal{E} \setminus \mathcal{Q}$ and $B \in \mathcal{Q}$. We claim that $\mathcal{Q}$ is an essential set of f. To see this, let us consider a pair of resolvable clauses $A, B \in \mathcal{I}(f)$ for which $C = R(A, B) \in \mathcal{Q}$. Since $C \in \mathcal{Q} \subseteq \mathcal{E}$ and since $\mathcal{E}$ is essential with no two of its clauses resolvable, we must have exactly one of A and B belong to $\mathcal{E}$, as we observed above. Say, we have $A \in \mathcal{E}$ and $B \in \mathcal{R}(\mathcal{X})$. Then, we have $C \in \mathcal{R}(\mathcal{X} \cup \{A\})$, and thus by the definition of $\mathbf{H}$ we must have (A, C) as an arc of $\mathbf{H}$. Since we assumed that $\mathcal{Q}$ is an initial set, with no arcs entering it, we must have $A \in \mathcal{Q}$, showing that $\mathcal{Q}$ is indeed essential.

Let us next consider all the initial strong components $\mathcal{Q}_1, \dots, \mathcal{Q}_k$ of $\mathbf{H}$. We claim that for any irredundant representation $\mathcal{C}$ of f we must have $|\mathcal{C} \cap \mathcal{Q}_j| = 1$ for $j = 1, \dots, k$ and $|\mathcal{C} \cap \mathcal{E}| = k$, from which the statement readily follows.

To see this, let us observe first that for all subsets $\mathcal{C} \subseteq \mathcal{I}(f)$ representing f we must have $\mathcal{C} \cap \mathcal{Q}_i \neq \emptyset$ for $i = 1, \dots, k$ by Theorem 6.6, since all the sets $\mathcal{Q}_i$, $i = 1, \dots, k$ are essential, as we observed above. Let us fix now an irredundant representation $\mathcal{C}$ of f, and let us choose clauses $C_j \in \mathcal{Q}_j \cap \mathcal{C}$ for $j = 1, \dots, k$, arbitrarily. Since $\mathcal{X}$ is exclusive we have $\mathcal{R}(\mathcal{X}) = \mathcal{R}(\mathcal{X} \cap \mathcal{C})$ by Lemma 5.4. Furthermore, since $\mathbf{H}$ is transitively closed, every clause $C \in \mathcal{E}$ is reachable by an arc from the set $\{C_1, \dots, C_k\}$, implying $\mathcal{E} \subseteq \mathcal{R}(\mathcal{X} \cup \{C_1, \dots, C_k\})$. Thus, by applying Lemma 4.4 we get $\mathcal{I}(f) = \mathcal{E} \cup \mathcal{R}(\mathcal{X}) \subseteq \mathcal{R}(\mathcal{X} \cup \{C_1, \dots, C_k\}) = \mathcal{R}((\mathcal{X} \cap \mathcal{C}) \cup \{C_1, \dots, C_k\}) \subseteq \mathcal{R}(\mathcal{C}) = \mathcal{I}(f)$, implying that $(\mathcal{X} \cap \mathcal{C}) \cup \{C_1, \dots, C_k\} \subseteq \mathcal{C}$ is a representation of f. Since $\mathcal{C}$ is assumed to be irredundant, equality follows, implying $\mathcal{C} \setminus \mathcal{X} = \mathcal{C} \cap \mathcal{E} = \{C_1, \dots, C_k\}$, from which the claim follows. □

Of course, it is clear that Theorem 6.16 implies the following corollary which more closely resembles the statement of Proposition 6.3.

Corollary 6.17. *Let f, $\mathcal{X}$, and $\mathcal{E}$ be as in the statement of Theorem 6.16, and let ϕ_1 and ϕ_2 be two distinct irredundant CNFs of f. Then $|\mathcal{C}(\phi_1) \cap \mathcal{E}| = |\mathcal{C}(\phi_2) \cap \mathcal{E}|$.*

Clearly, Proposition 6.3 is just a special case of Corollary 6.17 if we set $\mathcal{X}$ to be the set of all pure Horn clauses in $\mathcal{I}(f)$ (in this case $\mathcal{X} = \mathcal{R}(\mathcal{X})$) and $\mathcal{E}$ to be the set of all negative clauses in $\mathcal{I}(f)$.

Note that not every essential set $\mathcal{E}$ of implicates with no resolution inside has the properties claimed in Theorem 6.16. In other words, the condition that the complement of $\mathcal{E}$ is a resolution closure of an exclusive set cannot be neglected. A good example for this observation is the following: given any function f, its negative implicates in $\mathcal{I}(f)$ obviously form an essential set with no resolution inside. However, if f is not Horn, it may happen that the non-negative implicates in $\mathcal{I}(f)$ do not form a resolution closure of an exclusive set, and the properties claimed in Theorem 6.16 fail to hold. To see this, consider the following two CNFs:

$$\mathcal{C}_1 = (x \vee z)(\bar{x} \vee \bar{z})(x \vee \bar{y}),$$
$$\mathcal{C}_2 = (x \vee z)(\bar{x} \vee \bar{z})(\bar{y} \vee \bar{z}).$$

It is not hard to verify that they represent the same function and both are irredundant, but while $\mathcal{C}_1$ contains one negative clause, $\mathcal{C}_2$ contains two negative clauses.

7. Disjoint essential sets for Horn functions

In this section we shall restrict our attention to Horn functions only, in particular to the subclasses of quadratic Horn, acyclic Horn, and quasi-acyclic Horn functions. We shall show that Question 6.13 has an affirmative answer for all of the mentioned subclasses of Horn functions. We shall proceed as follows: after some simple preprocessing (getting rid of unit implicates) we shall use Theorem 6.16 to show that we can in fact concentrate only on pure Horn functions. Then we shall answer Question 6.13 for quadratic pure Horn and acyclic pure Horn functions. Finally, we will use a combination of these two results to answer Question 6.13 for quasi-acyclic pure Horn functions.

By standard Boolean terminology, a *unit* clause is a clause consisting of exactly one literal. If x or $\overline{x}$ is a unit prime implicate of a Boolean function f, then clearly no other prime implicates of f may contain the variable x (negated or not). This implies that also in $\mathcal{I}(f)$ the variable x appears only in the unit clause and nowhere else, which in turn means that this clause constitutes a trivial exclusive (and hence also essential) set (it cannot be derived by resolution from any other clauses in $\mathcal{I}(f)$).

It follows that any Horn function f can be decomposed into a conjunction of unit clauses f_1 and a Horn function f_2 which has no unit prime implicates, in such a way that f_1 and f_2 are defined on disjoint sets of variables, and $f = f_1 \wedge f_2$. Of course, Question 6.13 can be answered independently for f_1 and f_2 (due to the disjointness of their sets of variables) and the equality $\sigma(f) = \epsilon(f)$ trivially holds for f_1 by the above considerations. Therefore we can from now on restrict our attention (without loss of generality) solely to Horn functions with no unit prime implicates.

Let h be a Horn function and $\mathcal{C} \subseteq \mathcal{I}(h)$ a minimum (and therefore irredundant) set of clauses representing h such that $C_1, C_2, \dots, C_k$ are all the negative clauses in $\mathcal{C}$. Then Theorem 6.16 guarantees the existence of pairwise disjoint essential sets of negative implicates $\mathcal{Q}_1, \mathcal{Q}_2, \dots, \mathcal{Q}_k$ such that $C_j \in \mathcal{Q}_j$ for $j = 1, \dots, k$. Furthermore, if p is the pure Horn component of h (which is represented by the pure Horn clauses in C by Definition 5.6) then $\mathcal{I}(p)$ consists only of pure Horn clauses (since all prime implicates of a pure Horn function are pure Horn and a resolution of two pure Horn clauses is also pure Horn as recalled in Section 3), and thus $\mathcal{Q}_j \cap \mathcal{E} = \emptyset$ for every $j = 1, \dots, k$ and every subset $\mathcal{E}$ of $\mathcal{I}(p)$. This implies that if we manage to answer Question 6.13 for p by exhibiting pairwise disjoint essential subsets $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_\ell$ of $\mathcal{I}(p)$, where ℓ is the number of pure Horn clauses in $\mathcal{C}$, then the sets $\mathcal{E}_i, i = 1, \dots, \ell$, together with the sets $\mathcal{Q}_j, j = 1, \dots, k$, give an affirmative answer to Question 6.13 also for h.

Therefore we shall from now on restrict our attention to pure Horn functions (with no unit prime implicates) only. Let us start by introducing a very useful technique for verifying that a given clause is an implicate of a given pure Horn function.

Let η be a pure Horn CNF of a pure Horn function h. We shall define a *forward chaining* procedure which associates to any subset Q of the propositional variables of h a set M in the following way. The procedure takes as input the subset Q of propositional variables, initializes the set $M = Q$, and at each step it looks for a pure Horn clause $S \vee y$ in η such that $S \subseteq M$, and $y \notin M$. If such a clause is found, the propositional variable y is included into M, and the search is repeated as many times as possible.

In the relational database terminology the propositional variables in M are said to be "chained" to the subset Q (see e.g. [8]). In the expert systems terminology the usage of the clause $S \vee y$ is called "firing the rule" $\bigwedge_{x\in S} x \to y$ (see e.g. [11]).

```
FORWARD CHAINING PROCEDURE($\mathcal{C}$, $Q$)

Input:            A set $\mathcal{C}$ of pure Horn clauses, and
                  a subset $Q$ of propositional variables.
Initialization:   Set $M = Q$.
Main Step:        While $\exists\, C \in \mathcal{C} : Subg(C) \subseteq M$ and $Head(C) \notin M$
                  do $M = M \cup \{Head(C)\}$.
Stop:             Output $FC_{\mathcal{C}}(Q) = M$.
```

If the forward chaining procedure subsequently uses clauses $C_1, \dots, C_k$ (in this order) to enlarge the set M (starting with set Q), we say that the sequence of clauses $C_1, \dots, C_k$ forms a *forward chaining derivation* of $Head(C_k)$ from Q. The sequence is called an *irredundant* forward chaining derivation of $Head(C_k)$ from Q, if no proper subsequence of $C_1, \dots, C_k$ forms a forward chaining derivation of $Head(C_k)$ from Q. The following lemma [12,18], shows how the above procedure can help in determining whether a given clause is an implicate of a given CNF, or not.

Lemma 7.1. *Given a set $\mathcal{C}$ of pure Horn clauses, a subset Q of its propositional variables, and another one of its variables y, we have $y \in FC_{\mathcal{C}}(Q)$ if and only if $Q \vee y$ is an implicate of a function represented by $\mathcal{C}$.*

In what follows we will frequently not distinguish between CNFs and their sets of clauses, and thus for $\mathcal{C} = \mathcal{C}(\eta)$ we shall write both $FC_\eta(Q) = FC_{\mathcal{C}}(Q)$. If η' and η'' are two distinct pure Horn CNF representations of a given pure Horn function h and if Q is an arbitrary subset of the propositional variables, then by Lemma 7.1 $FC_{\eta'}(Q) = FC_{\eta''}(Q)$ because η' and η'' have the same set of implicates. Therefore, the set of propositional variables reachable from Q by forward chaining depends only on the underlying function rather than on a particular CNF representation. For this reason, we shall also use the expression $FC_h(Q)$ instead of $FC_\eta(Q)$ whenever we do not want to refer to a specific CNF.

Now let us return to essential sets of implicates. A key role in the upcoming proofs will be played by falsepoint sets $\mathcal{E}(t)$ from Definition 6.4 which were proved to be essential in Lemma 6.5. Using these sets we can show the following easy observation.

Lemma 7.2. *Let f be a Boolean function and let $\mathcal{C} = \{C_1, C_2, \ldots, C_m\} \subseteq \mathcal{I}(f)$ be an irredundant set of clauses representing f. Then for each $i = 1, \ldots, m$ there exists an essential set $\mathcal{E}_i$, for which $\mathcal{E}_i \cap \mathcal{C} = \{C_i\}$.*

Proof. Take $i \in \{1, \ldots, m\}$ arbitrarily. Since $\mathcal{C}$ is irredundant, there exists at least one Boolean vector t such that $C_i(t) = 0$ and $C_j(t) = 1$ for all $j \neq i$. Thus $\mathcal{E}_i$ can be set to $\mathcal{E}(t)$. □

It is clear that the sets $\mathcal{E}_i$, $i = 1, \ldots, m$, in the above proof need not be disjoint. However, we shall show that if f is a quadratic pure Horn, an acyclic pure Horn, or a quasi-acyclic pure Horn function and $\mathcal{C}$ is its minimum representation, then we can find Boolean vectors $t_1, \ldots, t_m$ such that the sets $\mathcal{E}_i = \mathcal{E}(t_i)$, $i = 1, \ldots, m$ in the above proof are pairwise disjoint, giving an affirmative answer to Question 6.13 for f.

In the remainder of this section we shall frequently use the implication graph $\mathbf{G}_f = (\mathbf{N}, \mathbf{A})$ of f defined in Section 3 (see Definition 3.3). Let us introduce some further notation. By K_x we shall denote that strong component of $\mathbf{G}_f$ which contains variable x. Le us denote by τ the partial ordering of strong components of $\mathbf{G}_f$ given by the arcs in $\mathbf{G}_f$, i.e., the existence of an arc in $\mathbf{G}_f$ from some variable in K_x to some variable in K_y is equivalent to the fact that $K_x <_\tau K_y$ whenever $K_x \neq K_y$. Finally, for a Boolean vector t (a truth assignment to the variables of f) and a variable x, $t[x]$ will denote the element of t which corresponds to x. Now we are ready to prove some useful properties connecting forward chaining and implication graphs.

Lemma 7.3. *Let $v \in FC_f(S)$ and let clauses $C_1, \ldots, C_k \subseteq \mathcal{I}(f)$ form an irredundant forward chaining derivation of v from S. Let x be an arbitrary variable used in clause C_i for some $1 \leq i \leq k$. Then either $x = v$ or (x, v) is an arc in $\mathbf{G}_f$.*

Proof. By the definition of an irredundant forward chaining derivation it follows that $Head(C_i) \in \cup_{j=i+1}^{k} Subg(C_j)$ for every $1 \leq i \leq k - 1$ and that $Head(C_k) = v$. By Lemma 3.4 we know that for every C_i each of its subgoals is connected by an arc in $\mathbf{G}_f$ to its head. A simple inductive argument going backward from C_k to C_i then shows that x is connected to v by a directed path (possibly of length zero if $x = Head(C_k) = v$) in $\mathbf{G}_f$, which together with the fact that $\mathbf{G}_f$ is a transitively closed graph finishes the proof. □

Lemma 7.4. *Let $\mathcal{C}$ be an irredundant and prime representation of a pure Horn function f. Let $C = A \vee x \in \mathcal{C}$ and $D = B \vee y \in \mathcal{I}(f)$. If $y \notin FC_{\mathcal{C}\setminus\{C\}}(B)$, then $A \subseteq FC_{\mathcal{C}\setminus\{C\}}(B)$ and either $x = y$ or (x, y) is an arc in $\mathbf{G}_f$.*

Proof. Since $D \in \mathcal{I}(f)$, Lemma 7.1 guarantees that $y \in FC_{\mathcal{C}}(B)$. Hence there exists an irredundant forward chaining derivation $C_1, \ldots, C_k$ of y from B using clauses from $\mathcal{C} \subseteq \mathcal{I}(f)$. Moreover, since $y \notin FC_{\mathcal{C}\setminus\{C\}}(B)$, each derivation of y from B must use C and thus $C_i = C$ for some $1 \leq i \leq k$ which directly implies $A \subseteq FC_{\mathcal{C}\setminus\{C\}}(B)$, and using Lemma 7.3 it also implies that either $x = y$ or (x, y) is an arc in $\mathbf{G}_f$. □

Lemma 7.5. *Let $\mathcal{C}$ be an irredundant and prime representation of a pure Horn function f, and let A,B be two sets of variables of f such that $A \subseteq FC_{\mathcal{C}}(B)$. Furthermore, let $A' \subseteq A$. Then $A' \subseteq FC_{\mathcal{C}}(B')$, where $B' = B \cap (A' \cup \{x \mid \exists a \in A' : (x, a) \text{ is an arc in } \mathbf{G}_f\})$.*

Proof. Let $a \in A'$ be arbitrary. Since $A \subseteq FC_{\mathcal{C}}(B)$ there exists an irredundant forward chaining derivation $C_1, \ldots, C_k$ of a from B using clauses from $\mathcal{C} \subseteq \mathcal{I}(f)$. By Lemma 7.3, every variable x used in $C_1, \ldots, C_k$ fulfils either $x = a$ or (x, a) is an arc in $\mathbf{G}_f$. Thus $a \in FC_{\mathcal{C}}(B \cap (\{a\} \cup \{x \mid (x, a) \text{ is an arc in } \mathbf{G}_f\}))$, from which the claim follows. □

Now we have all the necessary tools to answer Question 6.13 for the class of quadratic pure Horn functions.

Theorem 7.6. *Let f be a quadratic pure Horn function on n variables. Let m be the number of clauses in a minimum quadratic pure Horn CNF representing function f. Then there exist m pairwise disjoint essential sets of implicates of f.*

Proof. Consider a minimum set $\mathcal{C} = \{C_1, C_2, \ldots, C_m\} \subseteq \mathcal{I}(f)$ representing f (such a set can be constructed in polynomial time from any pure Horn representation of f either by the algorithm for a transitive reduction of a directed graph [20] or by the minimization algorithm for quasi-acyclic functions [14]). Recall that every prime implicate of a quadratic pure Horn function is a quadratic pure Horn clause, and a resolvent of two quadratic pure Horn clauses is again a quadratic pure Horn clause. It follows that not only $\mathcal{C}$ but also $\mathcal{I}(f)$ consists only of quadratic pure Horn clauses. Let us consider the implication graph $\mathbf{G}_f$ and the partial ordering τ of strong components of $\mathbf{G}_f$. There are two types of clauses in $\mathcal{C}$: for a clause $C_i = \overline{x} \vee y$ either $K_x <_\tau K_y$ (clause of type (A)) or $K_x = K_y$ (clause of type (B)). Given a clause $C_i = \overline{x} \vee y$ we define a set $\mathcal{E}_i$ in the following way:

$$\mathcal{E}_i = \begin{cases} \{(\overline{u} \vee v) \in I(f) \mid u \in K_x \text{ and } v \in K_y\} & \text{if } K_x <_\tau K_y \text{ (type (A))} \\ \{(\overline{z} \vee y) \in I(f) \mid z \in K_x = K_y \text{ and } z \neq y\} & \text{if } K_x = K_y \text{ (type (B))}. \end{cases}$$

If we think of quadratic pure Horn clauses as arcs in $\mathbf{G}_f$ then $\mathcal{E}_i$ of type (A) is a complete bipartite subgraph of all arcs going from K_x to K_y and $\mathcal{E}_i$ of type (B) is a star subgraph of all arcs in $K_x = K_y$ entering y.

Let us first show that the sets $\mathcal{E}_i$, $i = 1, \dots, m$ are pairwise disjoint sets of implicates of f. Obviously, a set of type (A) can never intersect a set of type (B). Two sets of type (A) intersect only if $\mathcal{C}$ contains two clauses $C_i = \overline{x} \vee y$, $C_j = \overline{u} \vee v$ of type (A) such that $K_x = K_u$ and $K_y = K_v$ (in fact, in such a case the sets $\mathcal{E}_i$ and $\mathcal{E}_j$ not only intersect but are equal). However, this is a contradiction to the irredundancy of $\mathcal{C}$, since C_i together with clauses of type (B) in $\mathcal{C}$ spanning K_x and K_y imply C_j. To see that no two sets of type (B) intersect we have to use the minimality of $\mathcal{C}$. Indeed, in a minimum representation each strong component of $\mathbf{G}_f$ must be spanned by a simple cycle of clauses (arcs) from $\mathcal{C}$ (see [20] or [14] for a proof of this simple fact). This means that for every y in a strong component of $\mathbf{G}_f$ of size larger than one, $\mathcal{C}$ contains exactly one clause (arc) of type B entering y. This in turn implies that all sets $\mathcal{E}_i$ of type B are pairwise disjoint.

It remains to show that each $\mathcal{E}_i$, $i = 1, \dots, m$, forms an essential set of clauses. For each C_i we shall define a Boolean vector $t_i \in \{0, 1\}^n$ and show that $\mathcal{E}_i = \mathcal{E}(t_i)$ for $i = 1, \dots, m$, which will finish the proof since each $\mathcal{E}(t_i)$ is essential by the proof of Theorem 6.6.

Let us start with the simpler case when $C_i = \overline{x} \vee y \in \mathcal{C}$ is of type (B). We define t_i in the following way:

$$t_i[z] = \begin{cases} 1 & \text{if } z \neq y \text{ and } (K_y =)K_x \leq_\tau K_z \\ 0 & \text{otherwise.} \end{cases}$$

Informally speaking, all variables in strong components “after” $K_x = K_y$ in order τ are assigned value 1, all variables in strong components “not after” $K_x = K_y$ in order τ are assigned value 0, and within $K_x = K_y$ only y is assigned value 0 while all other variables are assigned value 1. It is clear from the definition of t_i that $\mathcal{E}_i \subseteq \mathcal{E}(t_i)$, so we only have to show the opposite inclusion. Consider a clause $C \in \mathcal{E}(t_i)$. Since by definition $\mathcal{E}(t_i) \subseteq \mathcal{I}(f)$, C is quadratic pure Horn and we may write $C = \overline{u} \vee v$. Moreover, by Lemma 3.4 (u, v) is an arc in $\mathbf{G}_f$ and hence clearly $K_u \leq_\tau K_v$. The fact that $C(t_i) = 0$ implies $t_i[u] = 1$ and $t_i[v] = 0$. Hence $u \neq y$ and $K_x \leq_\tau K_u$ (all “ones” are in or “after” K_x). Because $K_u \leq_\tau K_v$, we have also that $K_x \leq_\tau K_v$. This together with fact that $t_i[v] = 0$ and definition of t_i implies that $v = y$. Putting all this together we get $K_y = K_x \leq_\tau K_u \leq_\tau K_v = K_y$, which implies $K_u = K_v = K_y = K_x$ and hence $C \in \mathcal{E}_i$.

Now let us consider that $C_i = \overline{x} \vee y \in \mathcal{C}$ is of type (A). This time we define t_i in the following way:

$$t_i[z] = \begin{cases} 0 & \text{if } K_z \neq K_x \text{ and } K_z \leq_\tau K_y \\ 1 & \text{otherwise.} \end{cases}$$

Note that (x, y) is an arc in $\mathbf{G}_f$, which implies $K_x <_\tau K_y$. Informally speaking, all variables in K_y and in strong components “before” K_y in order τ except for K_x are assigned value 0, while all variables in K_x and in strong components which are different from K_y and are “not before” K_y in order τ are assigned value 1. Again, it is clear from the definition of t_i that $\mathcal{E}_i \subseteq \mathcal{E}(t_i)$, so we only have to show the opposite inclusion. Let us take a clause $C \in \mathcal{E}(t_i)$. As in the previous case, C can be written as $C = \overline{u} \vee v$ where (u, v) is an arc in $\mathbf{G}_f$, $t_i[u] = 1$ and $t_i[v] = 0$. This assignment implies that $K_v \neq K_x$ and $K_v \leq_\tau K_y$. Since (u, v) is an arc in $\mathbf{G}_f$, $K_u \leq_\tau K_v \leq_\tau K_y$ and thus $K_u = K_x$ (since “ones” which are not in K_x are not before K_y). Therefore $K_x = K_u <_\tau K_v$. Proving that $K_v = K_y$ (and hence $C \in \mathcal{E}_i$) takes a bit more work. We have to show that no strong component which was assigned the value 0 (and thus could contain v) except for K_y is reachable by an arc from K_x. For this we shall use the irredundancy of $\mathcal{C}$. Let us by contradiction assume that $K_x = K_u <_\tau K_v <_\tau K_y$. By the definition of partial ordering τ we get that (v, y) is an arc in $\mathbf{G}_f$, and hence $\overline{v} \vee y$ is a prime implicate of f. Since $u \in K_x$, $\overline{x} \vee v$ is also a prime implicate of f. The following three observations now finish the proof:

- The fact $\overline{x} \vee v \in \mathcal{I}(f)$ implies $v \in FC_{\mathcal{C}}(\{x\})$ and also $v \in FC_{\mathcal{C}\setminus\{C_i\}}(\{x\})$ since otherwise Lemma 7.4 implies that either $y = v$ or (y, v) is an arc in $\mathbf{G}_f$, which is not possible since $K_v <_\tau K_y$.
- The fact $\overline{v} \vee y \in \mathcal{I}(f)$ implies $y \in FC_{\mathcal{C}}(\{v\})$ and also $y \in FC_{\mathcal{C}\setminus\{C_i\}}(\{v\})$ since otherwise Lemma 7.4 implies that $\{x\} \subseteq FC_{\mathcal{C}\setminus\{C_i\}}(\{v\})$, which in turn implies that there is a directed path from v to x in $\mathbf{G}_f$. Again, this is not possible since $K_x <_\tau K_v$.
- Putting the above two facts together gives $y \in FC_{\mathcal{C}\setminus\{C_i\}}(x)$ which is a contradiction to the irredundancy of $\mathcal{C}$. □

A similar result answering Question 6.13 can be derived for the class of acyclic pure Horn functions.

Theorem 7.7. *Let f be an acyclic pure Horn function on n variables. Let m be the number of clauses in the minimum acyclic pure Horn CNF representing function f. Then there exist m pairwise disjoint essential sets of implicates of f.*

Proof. The proof is in many ways similar to the proof of Theorem 7.6. This time, since all strong components of $\mathbf{G}_f$ are singletons, the clauses of type (B) do not exist, on the other hand clauses of type (A) are no longer restricted to quadratic ones.

Again consider a minimum prime set $\mathcal{C} = \{C_1, C_2, \dots, C_m\} \subseteq \mathcal{I}(f)$ representing f, which is (as proved in [12]) in fact the unique irredundant and prime CNF representing f. As in the previous proof we shall define for each $i = 1, \dots, m$ a Boolean vector $t_i \in \{0, 1\}^n$ and show that $C_i \in \mathcal{E}(t_i)$ and that the sets $\mathcal{E}(t_i)$, $i = 1, \dots, m$ are pairwise disjoint.

Since $\mathbf{G}_f$ is acyclic, the set of arcs of $\mathbf{G}_f$ induces a partial ordering of variables (vertices of $\mathbf{G}_f$) which we shall denote by τ, i.e., the existence of an arc (x, y) in $\mathbf{G}_f$ is equivalent to the fact, that $x <_\tau y$. Let us consider $C_i = X \vee y$ where $X = \{x_1, \dots, x_k\}$ and let t_i be defined as follows:

$$t_i[z] = \begin{cases} 0 & \text{if } z \notin FC_{\mathcal{C}\setminus\{C_i\}}(X) \text{ and } z \leq_\tau y \\ 1 & \text{otherwise .} \end{cases}$$

Note that this definition is a generalization of the corresponding definition (for a clause of type (A)) in the proof of Theorem 7.6. In the rest of this proof we shall proceed as follows. First we shall observe that $C_i \in \mathcal{E}(t_i)$. Then we shall show that $\mathcal{E}(t_i) \subseteq \{A \vee y \mid X \subseteq A\}$, which will make it easy to prove the disjointness of the sets $\mathcal{E}(t_i)$, $i = 1, \dots, m$.

Let us start by observing that $C_i(t_i) = 0$. Clearly, for each $j = 1, \dots, k$, $x_j \in FC_{\mathcal{C}\setminus\{C_i\}}(X)$ and $x_j \leq_\tau y$, hence $t_i[x_j] = 1$. Since $\mathcal{C}$ is irredundant, $y \notin FC_{\mathcal{C}\setminus\{C_i\}}(X)$ and hence $t_i[y] = 0$. By combining these two observations we get $C_i(t_i) = 0$ and therefore $C_i \in \mathcal{E}(t_i)$.

Now let us consider an arbitrary $C = A \vee b \in \mathcal{E}(t_i)$ where $A = \{a_1, \dots, a_\ell\}$. Let us show that $b = y$ and $X \subseteq A$. The fact that $C(t_i) = 0$ means that $t_i[a_j] = 1$ for $j = 1, \dots, \ell$ and $t_i[b] = 0$. By the definition of t_i this implies for every $j = 1, \dots, \ell$

$$a_j \in FC_{\mathcal{C}\setminus\{C_i\}}(X) \quad \text{or} \quad a_j \not\leq_\tau y \tag{i}$$

while

$$b \notin FC_{\mathcal{C}\setminus\{C_i\}}(X) \quad \text{and} \quad b \leq_\tau y. \tag{ii}$$

Since $C \in \mathcal{I}(f)$, Lemma 3.4 implies that (a_j, b) is an arc in $\mathbf{G}_f$ for each $j = 1, \dots, \ell$ and so $a_j <_\tau b \leq_\tau y$. This makes the option $a_j \not\leq_\tau y$ in (i) impossible and therefore

$$A \subseteq FC_{\mathcal{C}\setminus\{C_i\}}(X). \tag{iii}$$

Now (ii) and (iii) together clearly imply that

$$b \notin FC_{\mathcal{C}\setminus\{C_i\}}(A), \tag{iv}$$

since otherwise $b \in FC_{\mathcal{C}\setminus\{C_i\}}(X)$, which is not the case. From (iv) we get using Lemma 7.4 that

$$X \subseteq FC_{\mathcal{C}\setminus\{C_i\}}(A), \tag{v}$$

and also that either $y = b$ or (y, b) is an arc in $\mathbf{G}_f$ and hence $y \leq_\tau b$. However, in (ii) we have shown $b \leq_\tau y$ which together imply $y = b$.

Now we shall show that $X \subseteq A$. Let us assume by contradiction that there exists $x_j \notin A$ and let j be a minimal variable with respect to τ with this property. Let $A' \subseteq A$ and $X' \subseteq X$ be the sets of all variables less than x_j with respect to τ. Obviously, (iii) and Lemma 7.5 imply $A' \subseteq FC_{\mathcal{C}\setminus\{C_i\}}(X')$ which in turn implies $FC_{\mathcal{C}\setminus\{C_i\}}(A') \subseteq FC_{\mathcal{C}\setminus\{C_i\}}(X')$. Similarly, (v) and Lemma 7.5 imply $x_j \in FC_{\mathcal{C}\setminus\{C_i\}}(A')$. By Lemma 7.1 this means that $X' \vee x_j$ is an implicate of the function represented by $\mathcal{C} \setminus \{C_i\}$. But now the resolvent $(X \setminus \{x_j\}) \vee y$ of clauses C_i and $X' \vee x_j$ is an implicate of f contradicting the assumed primality of C_i. Hence $X \subseteq A$.

Now we know that when a clause C belongs to $\mathcal{E}(t_i)$, then it is of the form $A \vee y$ where $X \subseteq A$. Let us assume for the purpose of contradiction that $C \in \mathcal{E}(t_i) \cap \mathcal{E}(t_j)$ for some $i \neq j$. The only possibility is that $C_i = X_i \vee y$, $C_j = X_j \vee y$, and $X_i, X_j \subseteq A$. Moreover, we know from (iii) that $A \subseteq FC_{\mathcal{C}\setminus\{C_i\}}(X_i)$. Since $C_j \in \mathcal{C}$ and $C_j \neq C_i$ we have $y \in FC_{\mathcal{C}\setminus\{C_i\}}(X_j)$. But since $X_j \subseteq A$, we also have $y \in FC_{\mathcal{C}\setminus\{C_i\}}(A) \subseteq FC_{\mathcal{C}\setminus\{C_i\}}(X_i)$ which is a contradiction to irredundancy of $\mathcal{C}$, since C_i could be dropped without changing the function. □

It should be noted that if $\mathcal{C}$ is a quadratic pure Horn acyclic CNF representing a quadratic pure Horn acyclic function f, then it is not hard to observe that the proof of Theorem 7.7 shows that $\mathcal{E}_i = \mathcal{E}(t_i) = \{C_i\}$, since $\mathcal{I}(f)$ consists of only quadratic pure Horn clauses.

By combining the proofs of Theorems 7.6 and 7.7 we can now prove the same result for the class of quasi-acyclic pure Horn functions.

Corollary 7.8. *Let f be a quasi-acyclic pure Horn function. Let m be number of clauses in the minimum quasi-acyclic pure Horn CNF representing function f. Then there exist m pairwise disjoint essential sets of implicates of f.*

Proof. Let us consider a minimum prime set $\mathcal{C} = \{C_1, C_2, \dots, C_m\} \subseteq \mathcal{I}(f)$ representing f with the following properties:

- In each strong component Q of $\mathbf{G}_f$ one of its variables (denoted x_Q) is chosen as its "representative", and all clauses which contain variables from several different strong components (clauses of type A using the terminology of Theorem 7.6) contain no variables from Q except for x_Q.
- Each strong component Q of $\mathbf{G}_f$ of size k is spanned by k quadratic clauses from $\mathcal{C}$ which form a cycle (clauses of type B).

It was shown in [14] that there always exists a minimal CNF with the above properties. As in the previous proof we shall define for each $i = 1, \dots, m$ a Boolean vector $t_i \in \{0, 1\}^n$ and show that $C_i \in \mathcal{E}(t_i)$ and that the sets $\mathcal{E}(t_i)$, $i = 1, \dots, m$ are pairwise disjoint. We shall proceed as follows:

- If C_i is of type A we define t_i as in the proof of Theorem 7.7. To prove that $C_i \in \mathcal{E}(t_i)$ and the sets $\mathcal{E}(t_i)$ for all clauses of type A are disjoint, it suffices to follow line by line the proof of Theorem 7.7. The partial order used this time is a partial order of the strong components of $\mathbf{G}_f$ (or equivalently of the representative variables). The only difference is that the sets obtained by forward chaining include with every representative variable also all other variables in the given strong component (all its logically equivalent "copies"), but the proof remains valid.

- If C_i is of type B we define t_i as in the proof of Theorem 7.6 (for clauses of type B). Again, the fact that $C_i \in \mathcal{E}(t_i)$ and the sets $\mathcal{E}(t_i)$ for all clauses of type A are disjoint follows directly from the proof of Theorem 7.6. The proof uses the fact that $\mathcal{I}(f)$ contains only quadratic clauses. That is no longer true in the quasi-acyclic case, however what is true (and is sufficient for the validity of the proof) is that every clause in $\mathcal{I}(f)$ that contains a head and a subgoal from the same strong component of $\mathbf{G}_f$ is a quadratic clause (cannot contain any additional literals). Thus $\mathcal{E}(t_i)$ for a clause of type B consists (as before) only of quadratic clauses which represent arcs inside the strong component entering the head of C_i. This last observation also proves the "mixed" disjointness of sets $\mathcal{E}(t_i)$ for every pair of clauses of type A and B. □

8. Conclusions

The main results of this paper are presented in Sections 5–7. Section 5 introduces the notion of an exclusive set of implicates of a Boolean function and derives many properties that these sets possess. The most important property is proved in Theorem 5.5. Loosely speaking, given two different CNF representations $\mathcal{C}_1$ and $\mathcal{C}_2$ of a Boolean function f and an exclusive set $\mathcal{X}$ of implicates of f, the set of implicates in $\mathcal{C}_1$ that belong to $\mathcal{X}$ and the set of implicates in $\mathcal{C}_2$ that belong to $\mathcal{X}$ represent the same subfunction of f. Given $\mathcal{X}$, this subfunction is uniquely defined, and it is called the $\mathcal{X}$-component of f (or an exclusive component of f if $\mathcal{X}$ is clear from the context). The properties of exclusive components are summarized in Corollaries 5.7 and 5.8. The above results have a nice application in Boolean minimization. Indeed, if $\mathcal{X}$ is an exclusive set and $\mathcal{C}$ is the input CNF for the minimization problem, one can extract the sub-CNF which represents the $\mathcal{X}$-component of f, find its shortest CNF representation, and then insert this new sub-CNF back into the input CNF. That suggests a decomposition approach for minimization algorithms. Whenever one can find an exclusive subset of clauses of a given function or several pairwise disjoint exclusive subsets of clauses of a given function, it is possible to decompose the minimization problem, solve the subproblems separately, and then compose the obtained solutions back together. This approach is used by the authors of this paper in a manuscript (currently available as a research report [4]) for a polynomial time minimization of a subclass of Horn functions which properly includes the classes of quadratic, acyclic, and quasi-acyclic Horn functions.

Section 6 then introduces the notion of an essential set of implicates of a Boolean function. The main results presented in Theorems 6.6 and 6.15 state a nice duality (or orthogonality) between representations of a function f and essential sets of implicates of f. A set of clauses represents f if and only if it intersects every nonempty essential set of f. On the other hand, a set of clauses is essential only if it intersects all representations. Moreover, if a set of clauses intersects all representations and is minimal with this property, then it is essential. A simple corollary of these results provides the following lower bound on the length of CNF representations: given k pairwise disjoint nonempty essential sets of implicates of f, it is clear that every CNF representation of f contains at least k clauses. We pose a question (Question 6.13) for which classes of functions this lower bound is tight, i.e. for which classes of functions the number of clauses in a shortest representation always equals the maximum number of pairwise disjoint nonempty essential subsets of implicates. There are two natural open problems connected to this question:

(1) Are there any classes of functions for which $\epsilon(f) = \sigma(f)$ and computing this number is hard? Note that for all the classes we know for which $\epsilon(f) = \sigma(f)$ we can compute this number in polynomial time.
(2) What is the complexity of computing $\epsilon(f)$ in case $\epsilon(f) < \sigma(f)$?

Finally, in Section 7 we give an affirmative answer to Question 6.13 for the classes of quadratic, acyclic, and quasi-acyclic Horn functions. It should be noted that these results can be easily extended to the corresponding subclasses of renameable Horn functions. Given a Horn CNF $\mathcal{C}$, one can in linear time decide whether $\mathcal{C}$ is renameable Horn, and if so, output a set S of variables, such that switching (complementing) the variables in S produces a Horn CNF. In case this CNF falls into one of the above mentioned subclasses of Horn functions, one can find the appropriate Boolean vectors defining disjoint essential families as described in Section 7. After complementing the components of these vectors that correspond to the set S, one obtains disjoint essential sets of implicates of the original function. This observation also implies that Question 6.13 has an affirmative answer for the entire class of quadratic functions as it is well known that a quadratic CNF is either renameable Horn or identically zero, the latter being a trivial case in which there are no prime implicates and hence also no nonempty essential sets.

Acknowledgements

The authors would like to thank three anonymous referees for a thorough reading of the manuscript and for many helpful comments which helped to improve its final version. The second author gratefully acknowledges the support by the Czech Science Foundation (grant 201/07/0205). The fourth author gratefully acknowledges the support by the Czech Science Foundation (grant 201/07/P168).

References

[1] G. Ausiello, A. D'Atri, D. Sacca, Minimal representation of directed hypergraphs, SIAM Journal on Computing 15 (1986) 418–431.
[2] E. Boros, O. Čepek, On the complexity of Horn minimization, RUTCOR Research Report RRR 1-94, Rutgers University, New Brunswick, NJ, January 1994.
[3] E. Boros, O. Čepek, A. Kogan, Horn minimization by iterative decomposition, Annals of Mathematics and Artificial Intelligence 23 (1998) 321–343.

[4] E. Boros, O. Čepek, A. Kogan, P. Kučera, A subclass of Horn CNFs optimally compressible in polynomial time, RUTCOR Research Report RRR 11-2009, Rutgers University, New Brunswick, NJ, June 2008.
[5] H.K. Buning, T. Letterman, Propositional Logic: Deduction and Algorithms, Cambridge University Press, 1999.
[6] S.A. Cook, The complexity of theorem-proving procedures, in: Proceedings of the Third Annual ACM Symposium on theory of Computing, STOC'71, Shaker Heights, Ohio, United States, May 03–05, ACM, New York, NY, 1971, pp. 151–158.
[7] O. Čepek, Structural Properties and Minimization of Horn Boolean Functions, Doctoral Dissertation, Rutgers University, New Brunswick, NJ, October 1995.
[8] C. Delobel, R.G. Casey, Decomposition of a data base and the Theory of Boolean switching functions, IBM Journal of Research and Development 17 (1973) 374–386.
[9] M.R. Genesereth, N.J. Nilsson, Logical Foundations of Artificial Intelligence, Morgan Kaufmann, Los Altos, CA, 1987.
[10] P.L. Hammer, A. Kogan, Horn functions and their DNFs, Information Processing Letters 44 (1992) 23–29.
[11] P.L. Hammer, A. Kogan, Horn function minimization and knowledge compression in production rule bases, RUTCOR Research Report RRR 8-92, Rutgers University, New Brunswick, NJ, March 1992.
[12] P.L. Hammer, A. Kogan, Optimal compression of propositional Horn knowledge bases: Complexity and approximation, Artificial Intelligence 64 (1993) 131–145.
[13] P.L. Hammer, A. Kogan, Knowledge compression — logic minimization for expert systems, in: Computers As Our Better Partners, in: Proceedings of the IISF/ACM Japan International Symposium, World Scientific, Singapore, 1994, pp. 306–312.
[14] P.L. Hammer, A. Kogan, Quasi-acyclic propositional Horn knowledge bases: Optimal compression, IEEE Transactions on Knowledge and Data Engineering 7 (5) (1995) 751–762.
[15] D. Maier, Minimal covers in the relational database model, Journal of the ACM 27 (1980) 664–674.
[16] W. Quine, The problem of simplifying the truth functions, American Mathematical Monthly 59 (1952) 521–531.
[17] W. Quine, A way to simplify truth functions, American Mathematical Monthly 62 (1955) 627–631.
[18] S.J. Russel, P. Norvig, Artificial Intelligence: A Modern Approach, 2nd ed., Pearson Education, 2003.
[19] Petr Savický, Private communication.
[20] R.E. Tarjan, Depth first search and linear graph algorithms, SIAM Journal on Computing 2 (1972) 146–160.
[21] C. Umans, The minimum equivalent DNF problem and shortest implicants, Journal of Computer and System Sciences 63 (Dec.) (2001) 4.
[22] C. Umans, T. Villa, A.L. Sangiovanni-Vincentelli, Complexity of two-level logic minimization, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 25 (7) (2006) 1230–1246.

Ann Math Artif Intell (2009) 57:249–291
DOI 10.1007/s10472-010-9197-7

A subclass of Horn CNFs optimally compressible in polynomial time

Endre Boros · Ondřej Čepek · Alexander Kogan · Petr Kučera

Published online: 26 June 2010

Abstract The problem of Horn Minimization (HM) can be stated as follows: given a Horn CNF representing a Boolean function f, find a shortest possible (optimally compressed) CNF representation of f, i.e., a CNF representation of f which consists of the minimum possible number of clauses. This problem is the formalization of the problem of knowledge compression for speeding up queries to propositional Horn expert systems, and it is known to be NP-hard. There are two subclasses of Horn functions for which HM is known to be solvable in polynomial time: acyclic and quasi-acyclic Horn functions. In this paper we define a new class of Horn functions properly containing both of the known classes and design a polynomial time HM algorithm for this new class.

E. Boros · A. Kogan
RUTCOR, Rutgers University, P. O. Box 5062, New Brunswick, NJ 08903, USA

E. Boros
e-mail: boros@rutcor.rutgers.edu

O. Čepek (✉) · P. Kučera
Department of Theoretical Computer Science and Mathematical Logic, Charles University, Malostranské nám. 25, 118 00 Praha 1, Czech Republic
e-mail: cepek@ksi.ms.mff.cuni.cz

P. Kučera
e-mail: kucerap@ktiml.mff.cuni.cz

O. Čepek
Institute of Finance and Administration—VŠFS, Estonská 500, 100 00, Praha 10, Czech Republic
e-mail: ondrej.cepek@mff.cuni.cz

A. Kogan
Department of Accounting and Information Systems, Rutgers Business School, Rutgers University, Newark, NJ 07102, USA
e-mail: kogan@rutgers.edu

Keywords Horn functions · Boolean minimization · CQ functions · Essential sets · Exclusive sets

Mathematics Subject Classifications (2010) 06E30 · 94C10

1 Introduction

Horn functions are a very important subclass of Boolean functions. Their importance stems from the fact that the satisfiability problem (SAT), which is NP-complete for general Boolean formulae (see e.g. [13]) is solvable in linear time for Horn formulae [11, 22, 25]. This implies that certain real-life problems which require solving SAT become tractable if the underlying Boolean function in the problem is Horn. Such problems arise in several application areas, among others in artificial intelligence [9, 17, 18] and database design [10, 24]. Horn clauses constitute a very popular type of knowledge representation which is due to both the computational efficiency of reasoning and their relative richness (compared to other tractable classes) for capturing essential features of real-life problems.

In some applications an important problem is to find a shortest possible (i.e., optimally compressed) representation of a given Boolean function. For instance, in artificial intelligence this problem is equivalent to finding a most compact representation of a given knowledge base [17, 18]. Such transformation of a knowledge base accomplishes knowledge compression, since the actual knowledge does not change, while the size of the representation can be significantly reduced. The computational complexity of reasoning in the case of Horn knowledge bases is reduced accordingly, since the compressed knowledge base is guaranteed to remain Horn. The procedure of knowledge compression preprocesses the knowledge base, and can be done off-line. This results in speeding up on-line operation while answering queries. Therefore, the computational expense of a single run of knowledge compression will be quickly amortized over a large number of queries to the knowledge base.

Knowledge compression represents a special type of knowledge preprocessing. There are other knowledge preprocessing techniques such as knowledge compilation and knowledge condensation. All knowledge preprocessing methods utilize a trade-off between off-line and on-line processing, and are based on the observation that the additional computation resources consumed off-line will be compensated for by the ongoing reduction in computational effort during on-line operations. Knowledge compilation is a well developed type of knowledge preprocessing (see [6, 8, 23, 29, 30]). It was originally designed to construct Horn upper and lower bounds of a general Boolean function for subsequent use in answering queries. Some queries can be answered successfully and fast using only these bounds, while the attempt to answer such queries using the original Boolean function directly would be prohibitively expensive computationally. However, if the Horn bounds do not provide an answer, then the original function has to be used for answering such queries.

While knowledge compilation aims at reducing an intractable computational problem to a tractable one, knowledge compression focuses on Horn functions, for which the computational problem of answering queries is already tractable. However, the reduction of the total computational effort due to knowledge compression is nonetheless extremely important, since Horn functions used in practical applications can have very long representations. Such a situation can happen in many applications where Horn functions are generated automatically (as, e.g., when first order Horn theories are instantiated over finite but large domains). In these situations, the possible significant size reductions of Horn function representations enabled by knowledge compression are absolutely essential for saving computational resources and speeding up on-line query answering.

Another knowledge preprocessing technique is known as knowledge condensation (see [20, 21]). This technique attempts to identify the presence of functional dependencies implying that, in all models of a theory, the value of a variable is a function of values of some other variables. Then it may be possible to simplify a Horn function by repeatedly eliminating the variables whose values are determined by the values of other variables. The resulting "condensed" function may have much fewer variables and may be structurally simpler than the original function. Similarly to knowledge compilation, knowledge condensation changes not only the representation (as knowledge compression does), but the original function as well, while maintaining the possibility of using the condensed function for answering queries about the original one. Both knowledge condensation and knowledge compression can be used together with knowledge compilation to simplify the Horn bounds it produces.

Unfortunately, unlike satisfiability, the representation minimization problem is NP-hard not only in the general case, but also for Horn CNFs [1, 2, 7, 17, 24]. The Horn Minimization (HM) problem can be stated as follows: given a Horn CNF ϕ find a CNF ϕ' representing the same function and such that ϕ' consists of the minimum possible number of clauses. Paper [19] introduced two subclasses of Horn functions, acyclic and quasi-acyclic functions, for which HM is solvable in polynomial time.

In the present paper we shall introduce another subclass of Horn functions which properly contains both of the above subclasses and develop a polynomial time algorithm solving HM for the new class. The correctness of this algorithm heavily depends on nontrivial results about certain sets of implicates of Boolean functions proved in [4].

The practical significance of the optimal compression algorithm developed in this paper depends on whether the knowledge base is used for answering sufficiently many queries, and whether the response time is critically important or not. If the usage of the knowledge base is anticipated to be very infrequent, then it may not be justified to invest computational resources upfront in knowledge compression. If, on the other hand, the knowledge base is designed for active use, then the computational expense of a single run of knowledge compression will be abundantly compensated for by the ongoing computational savings derived in answering each subsequent query. Additionally, in those real-time applications where the response time in query answering is a major bottleneck, the proposed algorithm will be invaluable since it enables the reduction of the response time by up to a factor of 2, and this halving of the response time can make a difference between responding while the answer still matters and failing to do so.

The paper is structured as follows. In Section 2 we introduce the necessary notation and present several elementary results important for the subsequent presentation. In Section 3 we describe the classes of Horn functions for which HM is known to be solvable in polynomial time, and define a new class of Horn functions (component-wise quadratic or simply CQ functions), as well as prove some basic properties of this new class. Section 4 deals with so called exclusive and essential sets of implicates of (general) Boolean functions which were studied in [4]. We recall several key properties of these sets which are needed in the remainder of the paper. In Section 5 we return to the study of Horn functions. We associate with each Horn function two different directed graphs, show that these graphs define a nested structure of exclusive and essential sets of implicates of the function, and derive a series of technical statements about these sets. Finally, Section 6 contains the main result of this paper, namely the polynomial time HM algorithm for CQ functions, a proof of its correctness (which heavily depends on the results of the previous two sections), and an upper bound on its time complexity.

2 Basic notation, definitions, and results

In this section we shall introduce the necessary notation and summarize the basic known results that will be needed later in the text. The first subsection will present some basic facts about (general) Boolean functions and about the subclass of Horn functions. The second subsection will introduce a so called "forward chaining procedure" which constitutes a very useful tool for the study of Horn functions. Finally, the third subsection will present the standard graph terminology that will be used throughout this paper.

2.1 Boolean functions

A *Boolean function* f on n propositional variables $x_1, \ldots, x_n$ is a mapping $\{0, 1\}^n \rightarrow \{0, 1\}$. The propositional variables $x_1, \ldots, x_n$ and their negations $\overline{x}_1, \ldots, \overline{x}_n$ are called *literals* (*positive* and *negative literal*s, respectively). An elementary disjunction of literals

$$C = \bigvee_{i \in I} \overline{x}_i \vee \bigvee_{j \in J} x_j \tag{1}$$

is called a *clause*, if every propositional variable appears in it at most once, i.e., if $I \cap J = \emptyset$. A *degree* of a clause is the number of literals in it. For two Boolean functions f and g we write $f \leq g$ if

$$\forall (x_1, \ldots, x_n) \in \{0, 1\}^n \ : \ f(x_1, \ldots, x_n) = 1 \quad \Longrightarrow \quad g(x_1, \ldots, x_n) = 1 \tag{2}$$

Since each clause is in itself a Boolean function, formula (2) also defines the meaning of inequalities $C_1 \leq C_2$, $C_1 \leq f$, and $f \leq C_1$, where C_1, C_2 are clauses and f is a Boolean function.

We say that a clause C_1 *subsumes* another clause C_2 if $C_1 \leq C_2$ or, in other words, if the set of literals in C_1 is a subset of the set of literals in C_2 (e.g. the clause $\overline{x} \vee z$ subsumes the clause $\overline{x} \vee \overline{y} \vee z$). A clause C is called an *implicate* of a function f if $f \leq C$. An implicate C is called *prime* if there is no distinct implicate $C^{'}$ subsuming C, or in other words, an implicate of a function is prime if dropping any literal from it produces a clause which is not an implicate of that function.

It is a well-known fact that every Boolean function f can be represented by a conjunction of clauses (see e.g. [14]). Such an expression is called a *conjunctive normal form* (or CNF) of the Boolean function f. The special cases of the constants

1 and 0 are considered to be represented by the empty CNF (i.e., the one with no clauses) and by the CNF consisting of the empty clause (i.e., the one with no literals), respectively. It should be noted that a given Boolean function may have many different CNF representations. If two distinct CNFs, say ϕ_1 and ϕ_2, represent the same function, we say that they are *equivalent*, and denote this fact by $\phi_1 \equiv \phi_2$. A CNF ϕ representing a function f is called *prime* if each clause of ϕ is a prime implicate of the function f. A CNF ϕ representing a function f is called *irredundant* if dropping any clause from ϕ produces a CNF that does not represent f.

For example, in the CNF

$$(\overline{x}_1 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_2 \vee \overline{x}_3 \vee x_4) \wedge (\overline{x}_3 \vee \overline{x}_4)$$

the 2nd *clause can be dropped (although it is prime), and the* 4th *clause can be shortened (i.e., it is not prime). In fact, the same (Horn) function can be represented by the CNF*

$$(\overline{x}_1 \vee x_2) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_2 \vee \overline{x}_3) \wedge (\overline{x}_3 \vee \overline{x}_4)$$

which is both prime and irredundant.

The following two notational conventions will allow us to switch back and forth between sets of clauses and CNFs. For an arbitrary set of clauses $\mathcal{C}$ the symbol $\phi(\mathcal{C})$ denotes the CNF obtained by taking a conjunction of all clauses in $\mathcal{C}$. On the other hand, for an arbitrary CNF ϕ the symbol $\mathcal{C}(\phi)$ denotes the set of all clauses present in ϕ. We shall use the notion of "representing a given function" interchangeably for both CNFs and sets of clauses, i.e., if a CNF ϕ represents a function f we shall also say that the set of clauses $\mathcal{C}(\phi)$ represents f.

Two clauses C_1 and C_2 are said to be *resolvable* if they contain exactly one complementary pair of literals, i.e., if there exists exactly one propositional variable that appears uncomplemented in one of the clauses and complemented in the other. That means that we can write $C_1 = \tilde{C}_1 \vee x$ and $C_2 = \tilde{C}_2 \vee \overline{x}$ for some propositional variable x and clauses $\tilde{C}_1$ and $\tilde{C}_2$ which contain no complementary pair of literals. The clauses C_1 and C_2 are called *parent clauses* and the disjunction $R(C_1, C_2) = \tilde{C}_1 \vee \tilde{C}_2$ is called the *resolvent* of the parent clauses C_1 and C_2. Note that the resolvent is a clause (does not contain a propositional variable and its negation). The following is an easy lemma [5].

Lemma 1 *Let C_1 and C_2 be two resolvable implicates of a Boolean function f. Then $R(C_1, C_2)$ is also an implicate of f.*

We say, that a clause C can be *derived by a series of resolutions* from a CNF ϕ, if there exists a finite sequence $C_1, C_2, \ldots, C_p$ of clauses such that

(1) $C_p = C$, and
(2) for $i = 1, \ldots, p$, either $C_i \in \mathcal{C}(\phi)$ or there exist $j < i$ and $k < i$ such that $C_i = R(C_j, C_k)$.

Resolutions have a very important property (for consequence finding) usually called the *completeness of resolution*. Sometimes this property is also referred to as the Quine theorem after the author of one of the first papers in which this property was proved [26, 27], see also [5] for related material.

Theorem 1 *Let ϕ be a CNF representation of a Boolean function f and let C be a prime implicate of f. Then C can be derived from ϕ by a series of resolutions.*

Throughout this paper we shall also use the following notation. For an arbitrary set of clauses $\mathcal{C}$ the *resolution closure* of $\mathcal{C}$ denoted by $\mathcal{R}(\mathcal{C})$ is the set of all clauses obtainable through series of resolutions from the set $\mathcal{C}$ (allowing the resolvents to become parent clauses in subsequent resolutions).

For a Boolean function f let us denote by $\mathcal{I}^p(f)$ the set of its prime implicates, and let $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}^p(f))$. Note that not all implicates of f may belong to $\mathcal{I}(f)$. For instance, if f is defined by the CNF $\phi = (x_1 \vee x_2) \wedge (x_2 \vee x_3)$, then we have $\mathcal{I}(f) = \mathcal{I}^p(f) = \{(x_1 \vee x_2), (x_2 \vee x_3)\}$, however the clause $(x_1 \vee x_2 \vee x_3)$ is also an implicate of f.

For the purpose of measuring the complexity of algorithms we need ways of measuring the "size" of a given CNF ϕ. The following two definitions are the ones used most commonly in the literature:

– $|\phi|_c = \sum_{C \in \mathcal{C}(\phi)} 1$ (the number of clauses in ϕ),
– $|\phi|_\ell = \sum_{C \in \mathcal{C}(\phi)} |C|$ (the number of literals in ϕ),

where $|C|$ denotes the number of literals in clause C. For instance for the small CNF $\phi = (x_1 \vee x_2) \wedge (x_2 \vee x_3)$ of the above example, we have $|\phi|_c = 2$ and $|\phi|_\ell = 4$. Note also that $|\phi|_c = |\mathcal{C}(\phi)|$ if we use the standard notation $|S|$ to denote the cardinality of set S (recall that $\mathcal{C}(\phi)$ is the set of clauses in ϕ). We will swith back and forth betwen both notations in the subsequent text depending on whether we will be talking about CNFs or sets of clauses.

In this paper we shall mainly focus on the first measure, though many of our statements hold for the second measure, as well. Let us now turn our attention to the subclass of Boolean functions which is the focus of this paper, i.e., the class of Horn functions.

A clause C defined by (1) is called *negative* if it contains no positive literals (i.e., if $J = \emptyset$). It is called *pure Horn* (or in some literature *definite Horn*) if it contains exactly one positive literal (i.e., if $|J| = 1$). To simplify notation, we shall sometimes write a pure Horn clause $C = \bigvee_{x \in S} \overline{x} \vee y$ simply as $C = S \vee y$. Each propositional variable $x \in S$ is called a *subgoal of* C and the propositional variable y is called the *head of* C.[1] We shall denote $Head(C) = y$, $Subg(C) = S$, and $Vars(C) = S \cup \{y\}$.

A CNF is called *Horn* if it contains only negative and pure Horn clauses. A CNF is called *pure Horn* if it contains only pure Horn clauses. Finally, a Boolean function is called *Horn* if it has at least one representation by a Horn CNF, and similarly a Boolean function is called *pure Horn* if it has at least one representation by a pure Horn CNF.

It is known (see [15]) that each prime implicate of a Horn function is either negative or pure Horn, and each prime implicate of a pure Horn function is pure Horn. Thus, in particular, any prime CNF representing a Horn function is Horn, and any prime CNF representing a pure Horn function is pure Horn. The next statement was proved in [15].

[1]This terminology comes from the area of artificial intelligence, where the clause C is thought of as a "rule" $\bigwedge_{x \in S} x \longrightarrow y$.

Theorem 2 *Given a Horn CNF ϕ one can find in $O(|\phi|_\ell^2)$ time an irredundant and prime CNF ϕ' equivalent with ϕ.*

We shall use Theorem 2 as a (polynomial time) "preprocessing step" which will allow us to make an assumption that the CNF we work with (input CNF) is irredundant and prime (usually only primality will be needed).

2.2 Forward chaining procedure

In verifying that a given clause is an implicate of a given pure Horn function, a very useful and simple procedure is the following. Let η be a pure Horn CNF of a pure Horn function h. We shall define a *forward chaining* procedure which associates to any subset Q of the propositional variables of h a set M in the following way. The procedure takes as input the subset Q of propositional variables, initializes the set $M = Q$, and at each step it looks for a pure Horn clause $S \vee y$ in η such that $S \subseteq M$, and $y \notin M$. If such a clause is found, the propositional variable y is included into M, and the search is repeated as many times as possible.

In the relational database terminology the propositional variables in M are said to be "chained" to the subset Q (see e.g. [10]). In the expert system terminology the usage of the clause $S \vee y$ is called "firing the rule" $\bigwedge_{x \in S} x \to y$ (see e.g. [16]).

```
FORWARD CHAINING PROCEDURE($\mathcal{C}$, $Q$)

Input:            A set $\mathcal{C}$ of pure Horn clauses, and
                  a subset $Q$ of propositional variables.
Initialization:   Set $M = Q$.
Main step:        While $\exists\, C \in \mathcal{C} : Subg(C) \subseteq M$ and $Head(C) \notin M$
                  do $M = M \cup \{Head(C)\}$.
Stop:             Output $FC_{\mathcal{C}}(Q) = M$.
```

The following lemma, proved in [17], shows how the above procedure can help in determining whether a given clause is an implicate of a given CNF, or not.

Lemma 2 *Given a set $\mathcal{C}$ of pure Horn clauses, a subset Q of its propositional variables, and its variable $y \notin Q$, we have $y \in FC_{\mathcal{C}}(Q)$ if and only if $Q \vee y$ is an implicate of the function represented by $\mathcal{C}$.*

In what follows we will frequently refer to CNFs as well as their sets of clauses, and thus for $\mathcal{C} = \mathcal{C}(\eta)$ we shall write both $FC_\eta(Q) = FC_{\mathcal{C}}(Q)$. The following statement, proved in [19], shows that the forward chaining procedure can be efficiently implemented, i.e., that the complexity of performing the above mentioned verification is low.

Lemma 3 *Given a pure Horn CNF η and a subset Q of its propositional variables, the set $FC_\eta(Q)$ can be determined in $O(|\eta|_\ell)$ time.*

The complexity guaranteed by Lemma 3 is asymptotically the best possible one, since any implementation of the forward chaining procedure must at least read the entire input CNF. Let us conclude this section with a notational remark.

If η' and η'' are two distinct pure Horn CNF representations of a given pure Horn function h and if Q is an arbitrary subset of the propositional variables, then by Lemma 2 $FC_{\eta'}(Q) = FC_{\eta''}(Q)$ because η' and η'' have the same set of implicates. Therefore, the set of propositional variables reachable from Q by forward chaining depends only on the underlying function rather than on a particular CNF representation. For this reason, we shall also use the expression $FC_h(Q)$ instead of $FC_\eta(Q)$ whenever we do not want to refer to a specific CNF.

2.3 Implication graphs of Horn functions

Let us recall first some standard notions from graph theory. A *directed graph* (sometimes abbreviated to *digraph*) is an ordered pair $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ where $\mathbf{N}$ is the set of *nodes* and $\mathbf{A}$ is the set of *arcs*, and where an arc is an ordered pair of nodes.

A *directed path* is a nonempty sequence of arcs $a_1, a_2, ..., a_p$ such that $a_i = (x_i, x_{i+1})$ for some vertices $x_1, x_2, ..., x_{p+1}$. A *cycle* is a path such that $x_1 = x_{p+1}$. A directed graph is called *strongly connected* if for any two nodes x and y there exist both a directed path from x to y and a directed path from y to x. If a graph $\mathbf{D}$ is not strongly connected then its vertex set can be decomposed in a unique way into maximal strongly connected subsets, called the *strong components* of $\mathbf{D}$.

A subset $X \subseteq \mathbf{N}$ of nodes is called an *initial set* of $\mathbf{D}$ if $(u, v) \in \mathbf{A}$ and $v \in X$ imply $u \in X$, and it is called a *terminal set* of $\mathbf{D}$ if $(u, v) \in \mathbf{A}$ and $u \in X$ imply $v \in X$. We shall denote by $Cone_{\mathbf{D}}(X)$ the smallest (with respect to inclusion) initial set of $\mathbf{D}$ that contains X, and by $Anticone_{\mathbf{D}}(X)$ the smallest (with respect to inclusion) terminal set of $\mathbf{D}$ that contains X.

A directed graph is called *acyclic* if it contains no directed cycle. Note that in such a case every strong component consists of a single node. If $\mathbf{D}$ is an acyclic directed graph with a node set $\mathbf{N} = \{x_1, \ldots, x_n\}$, then an ordering of the nodes $(x_{i_1}, \ldots, x_{i_n})$ is called a *topological order* on $\mathbf{N}$ if for every arc $(x_{i_j}, x_{i_k}) \in \mathbf{A}$ we have $i_j < i_k$.

If $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ is a directed graph, then the *transitive closure* of $\mathbf{D}$ is a directed graph $\overline{\mathbf{D}} = (\mathbf{N}, \overline{\mathbf{A}})$ where $(x, y) \in \overline{\mathbf{A}}$, whenever there is a directed path from x to y in the digraph $\mathbf{D}$. For each directed graphs $\mathbf{D}$ its transitive closure $\overline{\mathbf{D}}$ is uniquely defined. A *transitive reduction* of $\mathbf{D}$ is a graph $\mathbf{D}_R = (\mathbf{N}, \mathbf{A}_R)$ such that $\overline{\mathbf{D}}_R = \overline{\mathbf{D}}$ (the digraphs $\mathbf{D}_R$ and $\mathbf{D}$ have the same transitive closure) and $|\mathbf{A}_R|$ is minimum. As opposed to the transitive closure, there may be several distinct transitive reductions of the same digraph $\mathbf{D}$.

Finally, if $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ is a directed graph with strong components $C_1, \ldots, C_s$, then the directed graph $\mathbf{D}' = (\mathbf{N}', \mathbf{A}')$ on the set of nodes $\mathbf{N}' = \{C_1, \ldots, C_s\}$ with arcs

$$(C_i, C_j) \in \mathbf{A}' \text{ iff } \exists x \in C_i \ \exists y \in C_j \text{ such that } (x, y) \in \mathbf{A}$$

is called the *acyclic condensation* of the digraph $\mathbf{D}$.

Let us note that the strong components of a directed graph $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ along with the topological order of its acyclic condensation can be found in $O(|\mathbf{A}|)$ time [31].

Let us recall next some very useful definitions from [19], associating directed graphs to Horn CNFs and Horn functions.

Definition 1 For a Horn CNF ϕ let $\mathbf{G}_\phi = (\mathbf{N}, \mathbf{A}_\phi)$ be the digraph defined by

$$\mathbf{N} = \{x \mid x \text{ is a propositional variable in } \phi\}$$

$$\mathbf{A}_\phi = \{(x, y) \mid \exists \text{ a clause } C \in \mathcal{C}(\phi) \text{ such that } C \geq \overline{x} \vee y\}.$$

In other words, for each pure Horn clause C in ϕ, the graph $\mathbf{G}_\phi$ contains as many arcs as is the number of subgoals in C, with each arc going from the corresponding subgoal to the head of C. Since a Horn function can be represented by several different Horn CNFs, seemingly we can associate in this way several different graphs to a Horn function. However, as it was shown in [3], all these graphs share several important features. Before stating these features formally, let us consider a small example.

Example 1 Let us consider the following CNF

$$\phi = (\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge (\overline{x}_2 \vee x_1) \wedge (\overline{x}_3 \vee x_2) \wedge$$

$$\wedge (\overline{x}_5 \vee x_4) \wedge (\overline{x}_5 \vee x_6) \wedge (\overline{x}_6 \vee x_7) \wedge (\overline{x}_1 \vee \overline{x}_4 \vee x_5) \wedge (\overline{x}_2 \vee \overline{x}_6 \vee x_5) \wedge (\overline{x}_3 \vee \overline{x}_7 \vee x_6)$$

It is a well known fact that pure Horn CNFs are in a one to one correspondence with directed hypergraphs, where the propositional variables play the role of the nodes of the the hypergraph and each clause corresponds to a directed hyperedge. Each such directed hyperedge is directed from the set of negative literals (subgoals) in the clause to the unique positive literal (head) of the clause. Figure 1 depicts the hypergraph that corresponds to the above defined CNF ϕ.

By definition, the directed graph $\mathbf{G}_\phi$ can be obtained from the directed hypergraph of ϕ by replacing every directed hyperedge by a set of directed edges, one edge per subgoal. Figure 2 depicts $\mathbf{G}_\phi$ corresponding to the CNF ϕ given in Example 1. Note that $\mathbf{G}_\phi$ consists of two strong components denoted by C_1 and C_2.

Since we will be using the above CNF ϕ and the function f defined by ϕ as a running example throughout the rest of this paper, let us also consider the set of prime implicates of f. It is not hard to see that $\mathcal{I}^p(f)$ consists of all six quadratic clauses on the set $\{x_1, x_2, x_3\}$, i.e., of the clauses

$$(\overline{x}_1 \vee x_2), (\overline{x}_2 \vee x_3), (\overline{x}_2 \vee x_1), (\overline{x}_3 \vee x_2), (\overline{x}_3 \vee x_1), (\overline{x}_1 \vee x_3)$$

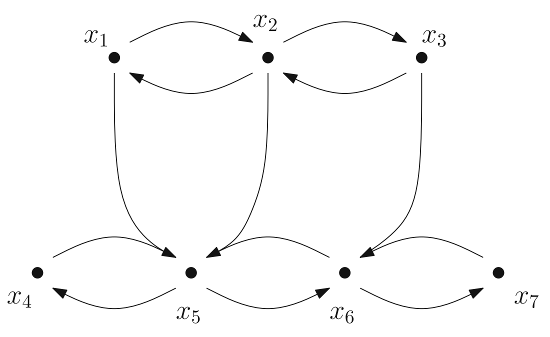

Fig. 1 Hypergraph corresponding to CNF ϕ from Example 1

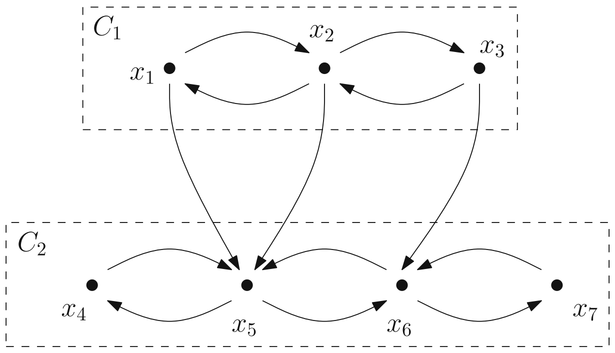

Fig. 2 CNF graph G_φ corresponding to CNF ϕ from Example 1

plus the three quadratic clauses on the set $\{x_4, x_5, x_6, x_7\}$ which are explicitly present in ϕ and one obtained by transitivity, i.e., of the clauses

$$(\overline{x}_5 \vee x_4), (\overline{x}_5 \vee x_6), (\overline{x}_6 \vee x_7), (\overline{x}_5 \vee x_7)$$

plus all cubic clauses of the form $(\overline{x}_i \vee \overline{x}_j \vee x_k)$ where $i \in \{1, 2, 3\}$, $j, k \in \{4, 5, 6, 7\}$, $j \neq k$, and where $(\overline{x}_j \vee x_k)$ is not a prime implicate of f, namely the cubic clauses

$$\begin{array}{l}(\overline{x}_i \vee \overline{x}_4 \vee x_5),\ (\overline{x}_i \vee \overline{x}_4 \vee x_6),\ (\overline{x}_i \vee \overline{x}_4 \vee x_7),\\ (\overline{x}_i \vee \overline{x}_6 \vee x_4),\ (\overline{x}_i \vee \overline{x}_6 \vee x_5),\\ (\overline{x}_i \vee \overline{x}_7 \vee x_4),\ (\overline{x}_i \vee \overline{x}_7 \vee x_5),\ (\overline{x}_i \vee \overline{x}_7 \vee x_6),\end{array}$$

for $i \in \{1, 2, 3\}$. Finally, the set $\mathcal{I}(f)$ additionally contains those cubic clauses of the form $(\overline{x}_i \vee \overline{x}_j \vee x_k)$ where $i \in \{1, 2, 3\}$, $j, k \in \{4, 5, 6, 7\}$, $i \neq k$, and where $(\overline{x}_j \vee x_k)$ is a prime implicate; it also contains all 36 (non-prime) degree four clauses of the form $(\overline{x}_i \vee \overline{x}_\ell \vee \overline{x}_j \vee x_k)$ where $i, \ell \in \{1, 2, 3\}$, $i \neq \ell$, $j, k \in \{4, 5, 6, 7\}$, $j \neq k$, and all 12 (non-prime) degree five clauses of the form $(\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_j \vee x_k)$ where $j, k \in \{4, 5, 6, 7\}$, $j \neq k$.

Theorem 3 [3] *Let ϕ_1 and ϕ_2 be two distinct prime CNFs representing the same Horn function f and let x, y be arbitrary propositional variables from f. Then there is a directed path from x to y in $\mathbf{G}_{\phi_1}$ if and only if there is a directed path from x to y in $\mathbf{G}_{\phi_2}$. Moreover, it then follows that $\mathbf{G}_{\phi_1}$ and $\mathbf{G}_{\phi_2}$ have identical transitive closures, identical strong components, and identical acyclic condensations.*

Theorem 3 allows us to associate a graph directly to a Horn function rather than to its particular Horn CNF representations.

Definition 2 Let f be a Horn function and ϕ its arbitrary prime CNF representation. Then we define $\mathbf{G}_f$ as the transitive closure of $\mathbf{G}_\phi$.

Theorem 3 suggests that for a given Horn function f the strong components of $\mathbf{G}_f$ play an important role in how the set of all prime CNF representations of f are structured. In what follows we shall call $\mathbf{G}_\phi$ and $\mathbf{G}_f$ the *implication graph*s of ϕ and f, respectively.

Note that the correspondence between pure Horn CNFs or functions and their implication graphs is not one to one. Given a CNF or a function, its implication graph is uniquely defined but not vice versa. For instance, if we replace the cubic clause $(\overline{x}_1 \vee \overline{x}_4 \vee x_5)$ by two quadratic clauses $(\overline{x}_1 \vee x_5)$ and $(\overline{x}_4 \vee x_5)$ in Example 1 then the new CNF represents a different pure Horn function than the original CNF ϕ but the implication graphs of both CNFs are identical and the same is true for the implication graphs of both functions.

Let us state finally an important property of implication graphs, slightly generalizing a statement of [19].

Lemma 4 *Let h be a pure Horn function, and let $C \in \mathcal{I}(h)$. Then $(x, Head(C))$ is an arc in $\mathbf{G}_h = (\mathbf{N}, \mathbf{A})$ for every $x \in Subg(C)$.*

Proof This statement was shown in [19] for all prime implicates $C \in \mathcal{I}^p(h)$. Since $\mathcal{I}(h) = \mathcal{R}(\mathcal{I}^p(h))$, every implicate $C \in \mathcal{I}(h)$ can be derived by a series of resolutions from prime implicates of h. Thus, the statement will follow by the inductive use of the following argument:

Let $C_1 = B_1 \vee x$ and $C_2 = B_2 \vee \overline{x} \vee y$ be two resolvable clauses such that $(z, x) \in \mathbf{A}$ for every $z \in B_1$ and $(z, y) \in \mathbf{A}$ for every $z \in B_2 \cup \{x\}$. Then, the clause $C = R(C_1, C_2) = B_1 \vee B_2 \vee y$ has the property that $(z, y) \in \mathbf{A}$ for every $z \in B_1 \cup B_2$, since $(z, y) \in \mathbf{A}$ is assumed for $z \in B_2$ and since $\mathbf{G}_h$ is transitively closed and $(x, y) \in \mathbf{A}$ and $(z, x) \in \mathbf{A}$ for all $z \in B_1$ by our assumptions. $\square$

3 Polynomially solvable cases of HM

The notion of an implication graph of a Horn function f carries a lot of information about the CNF representations of f, allowing the characterization of important special classes for which HM is polynomially solvable.

A Horn CNF ϕ is said to be *acyclic* if its associated implication graph $\mathbf{G}_\phi$ is acyclic. A Horn function f is called *acyclic* if it admits at least one acyclic CNF representation. It was shown in [19] that every acyclic function has a unique irredundant and prime representation, implying that this unique CNF also constitutes a minimal representation of the given function with respect to both complexity measures we introduced. Hence the "preprocessing phase" corresponding to Theorem 2 (i.e., transforming the input CNF into an irredundant and prime one) itself represents a polynomial time HM algorithm for acyclic functions.

Let us call two propositional variables x and y *logically equivalent* in a Horn function f if the clauses $\overline{x} \vee y$ and $\overline{y} \vee x$ are implicates of f. A Horn CNF ϕ is then said to be *quasi-acyclic* (see [19]) if every strong component of its associated implication graph $\mathbf{G}_\phi$ consists of a set of logically equivalent propositional variables. A Horn function f is called *quasi-acyclic* if it admits at least one quasi-acyclic CNF representation.

Note that every acyclic CNF ϕ is also quasi-acyclic since each strong component of $\mathbf{G}_\phi$ is a singleton. The name quasi-acyclic comes from the fact that picking a representative in each set of logically equivalent propositional variables and substituting this representative for all the other logically equivalent variables in the set results in an acyclic CNF (i.e., the CNF is essentially acyclic except for the fact that each variable can have several "names"). In order to understand the structure of quasi-acyclic functions it is important to realize that if f is a quasi-acyclic function and

x, y are propositional variables from the same strong component of $\mathbf{G}_f$ then, since both $\overline{x} \vee y$ and $\overline{y} \vee x$ are implicates of f, no prime pure Horn implicate of f with degree three or more may contain a subgoal from the same strong component of $\mathbf{G}_f$ as the head. This means that the pure Horn clauses in any prime CNF representation of f can be partitioned into two groups. The first group (let us call it group A) contains clauses where all the subgoals are in different strong component(s) of $\mathbf{G}_f$ than the head, while the second group (group B) contains quadratic clauses with both the subgoal and the head belonging to the same strong component of $\mathbf{G}_f$. Loosely speaking, the clauses in group B "generate" the strong components of $\mathbf{G}_f$ while the clauses in group A "generate" its acyclic condensation. It was proved in [19] that HM can be solved in polynomial time for quasi-acyclic functions.

Generalizing further the above classes (still using the implication graph) leads us to the main concept of this paper.

Definition 3 Let us call a pure Horn clause C *component-wise quadratic* (or CQ for short) with respect to a Horn function f if at most one subgoal of C belongs to the strong component of $\mathbf{G}_f$ containing the head of C. A Horn CNF ϕ representing a function f is said to be CQ if every pure Horn clause of ϕ is CQ with respect to f. Finally, a Horn function f is called CQ if it admits at least one CQ CNF representation.

The intuition behind the name, component-wise quadratic, is that if we restrict any such CNF to variables from a single strong component of its implication graph, the pure Horn part of the resulting CNF is always quadratic. The negative part may contain clauses of higher degrees than two, but as we shall see later in this paper, negative clauses play no essential role in the Horn minimization problem. Note that the CNF ϕ introduced in Example 1 is a CQ Horn CNF. It consists of seven quadratic and three cubic clauses. The quadratic clauses in ϕ are of course CQ clauses. It can be easily seen from Fig. 2 that each of the three cubic clauses in ϕ has its head together with one subgoal in strong component C_2 while the other subgoal is in C_1. Therefore all three cubic clauses in ϕ are CQ clauses as well.

Once again, it is possible to ask what happens if a CQ function is represented by a CNF which is not CQ. Fortunately, as in the acyclic and quasi-acyclic cases, it is enough to do the preprocessing phase to arrive to a CQ representation, as the following statement shows.

Theorem 4 *Let f be a CQ function. Then any prime CNF representation of f is CQ.*

Proof By Definition 3 there exists a CQ CNF ϕ which represents f. Let us replace every clause in ϕ by a prime implicate of f which subsumes the given clause and let us denote the resulting prime CNF of f by ϕ'. Note that each pure Horn clause C of ϕ' has two properties:

1. C is a CQ clause;[2]
2. there is an arc in $\mathbf{G}_f$ from every subgoal of C to the head of C.

[2]Here we mean CQ with respect to f. Whenever it is obvious which function is meant, we shall omit referring to it in the subsequent text.

The first property follows from the fact that deleting literals from a CQ clause must yield another CQ clause. The second property follows from the fact that $\mathbf{G}_f$ is the transitive closure of $\mathbf{G}_{\phi'}$ by Definition 2 (here we need primality and that is the reason why we transformed ϕ into ϕ'). We shall show that a resolvent of any two clauses which satisfy the above two properties has again both of these properties.

So let $C_1 = A \vee x$ and $C_2 = B \vee \bar{x} \vee y$ be two arbitrary resolvable clauses belonging to $\mathcal{I}(f)$ and satisfying the above two properties. Let us denote their resolvent by $C = A \vee B \vee y$, and observe that $C \in \mathcal{I}(f)$ by Lemma 1 and by the definition of $\mathcal{I}(f)$. Thus, the second property follows by Lemma 4.

To verify the first property, let us consider the strong components S_x and S_y of $\mathbf{G}_f$ which contain x and y respectively.

– If $S_x \neq S_y$, then S_x precedes S_y in the partial order imposed by the acyclic condensation of $\mathbf{G}_f$ (because there is an arc from x to y in $\mathbf{G}_f$), and hence so do all strong components which have a nonempty intersection with the set A (because for every $z \in A$ there is an arc from z to x in G_f). Hence $A \cap S_y = \emptyset$. Now the fact that C_2 is CQ implies $|B \cap S_y| \leq 1$ and thus $|(A \cup B) \cap S_y| \leq 1$ follows, implying that C is CQ.
– If $S_x = S_y$, then $\{x, y\} \subseteq S_y$, and thus the fact that C_1 and C_2 are CQ implies $|A \cap S_y| \leq 1$ and $B \cap S_y = \emptyset$. Therefore again $|(A \cup B) \cap S_y| \leq 1$ follows, proving that C is CQ.

By completeness of resolution (Theorem 1) every prime implicate of f can be derived from ϕ' by a series of resolutions. This implies that every prime implicate of f satisfies the above two properties, i.e., in particular every prime implicate of f is a CQ clause, which completes the proof. □

Note that the above proof actually implies a bit more: if f is a CQ function then not only all prime implicates of f are CQ but even all clauses in $\mathcal{I}(f)$ are CQ. Let us formulate this easy observation as a corollary.

Corollary 1 *Let f be a CQ function and C an arbitrary clause in $\mathcal{I}(f)$. Then C is CQ with respect to f.*

Of course, Theorem 4 also immediately implies the following corollary concerning the recognition of CQ functions.

Corollary 2 *Let ϕ be a Horn CNF. Then it can be checked in $O(|\phi|_\ell^2)$ time whether ϕ represents a CQ function or not.*

Proof Using Theorem 2 we can transform ϕ into a logically equivalent irredundant and prime CNF ϕ' in $O(|\phi|_\ell^2)$ time. Note that the new CNF is at most as long as the input one, i.e., $|\phi'|_\ell \leq |\phi|_\ell$. Now by Theorem 4 we get that the represented function is CQ if and only if ϕ' is a CQ CNF, which can be checked in $O(|\phi'|_\ell)$ time (both building the implication graph and detecting its strong components as well as checking that every clause fulfils the CQ property takes linear time in the length of ϕ'). □

The main aim of this paper is to show that HM is polynomially solvable for CQ-functions.

Theorem 5 *Let h be a CQ-function on n variables, represented by an irredundant and prime CNF $\mathcal{C}$ consisting of m clauses and ℓ literals. Then, a minimum CNF representation $\mathcal{C}^*$ of h can be found in $O(n^2 + m\ell)$ time.*

In the rest of the paper we shall present a decomposition based proof for the above statement. To arrive at such a proof, we first need to analyze the structure of potentially useful decompositions. We accomplish this by studying the structure of certain subfamilies of implicates of a Boolean function (in general, we do not restrict ourselves to Horn functions only). In particular, in Section 4 we consider subfamilies of implicates which define subfunctions, the representation of which can be chosen independently from the other clauses of the considered CNF representation. We also consider subfamilies from which every CNF representation must contain some clauses, leading to a min-max relation. In Section 5 we return to Horn functions, and provide tools to identify the above mentioned useful subfamilies of implicates in a constructive way. The main tool in this will be a new graph associated to a Horn function h, the vertices of which are the implicates of h, and which we shall call the *clause graph* of h. Finally, in Section 6 we present an algorithm for HM, and prove its correctness and complexity, as claimed in Theorem 5.

A natural question to ask is whether it is possible to further extend the class of CQ functions by allowing the pure Horn clauses to have not at most one but at most two (three, four, etc.) subgoals in the same strong component as the head, and still maintain the polynomial time solvability of HM. The answer is no unless P=NP. The reason is that even allowing just two subgoals to fall in the same strong component as the head would include all cubic Horn functions into the extended class (a Horn function is cubic if it admits a CNF representation with the highest clause degree at most three). However, it was proved in [2, 7] that HM is NP-hard for cubic Horn CNFs.

4 Exclusive and essential sets of implicates

In this section we recall properties of CNF representations of Boolean functions proved in [4]. These properties are applicable to all Boolean functions not just Horn ones. We also state and prove a decomposition lemma which is a consequence of the recalled properties. Later in the paper we shall return to Horn functions, and more specifically to CQ-functions, and apply the general results developed in this section.

In the remainder of this section let us consider an arbitrary but fixed Boolean function f, the set $\mathcal{I}^p(f)$ of all prime implicates of f, and the set $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}^p(f))$ of all implicates of f that can be generated from the prime implicates of f by series of resolutions. Note that $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}(f))$, i.e., the set $\mathcal{I}(f)$ is closed under resolution.

Let us start by recalling two simple technical lemmas from [4] (appearing there as Lemmas 4.3 and 4.4) which deal with properties of resolution closures of sets of clauses.

Lemma 5 *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two sets of clauses. Then $\mathcal{R}(\mathcal{C}_1) = \mathcal{R}(\mathcal{C}_2)$ implies that $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{C}_2)$, i.e., if the sets have the same resolution closure then they represent the same function.*

Lemma 6 *Let $\mathcal{X}, \mathcal{Z} \subseteq \mathcal{I}(f)$ be two arbitrary sets of clauses. Then $\mathcal{R}(\mathcal{X} \cup \mathcal{Z}) = \mathcal{R}(\mathcal{X} \cup \mathcal{R}(\mathcal{Z}))$.*

4.1 Exclusive components of functions

Let us now define the first key concept of this section, which helps us to decompose the problem of Horn minimization.

Definition 4 Given a set $\mathcal{C}$ of clauses, a subset $\mathcal{X} \subseteq \mathcal{C}$ is called an *exclusive subset of* $\mathcal{C}$ if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{C}$ the following implication holds:

$$R(C_1, C_2) \in \mathcal{X} \Longrightarrow C_1 \in \mathcal{X} \text{ and } C_2 \in \mathcal{X},$$

i.e., the resolvent belongs to $\mathcal{X}$ only if both parent clauses are in $\mathcal{X}$. In particular, if $\mathcal{C} = \mathcal{I}(f)$ for a Boolean function f, we call such a subset $\mathcal{X}$ an exclusive set of clauses of f (or simply an exclusive set, if f or $\mathcal{C}$ is clear from the context).

Let us first claim in the next lemma [4] some simple properties possessed by exclusive sets. Since all these properties follow directly from Definition 4 we shall omit the proofs.

Lemma 7 *Let $\mathcal{C}$ be an arbitrary set of clauses. Then,*

(a) *if $\mathcal{A}$ is an exclusive subset of $\mathcal{B}$ and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{A}$ is an exclusive subset of $\mathcal{C}$;*
(b) *if $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{C}$, and $\mathcal{A}$ is an exclusive subset of $\mathcal{C}$, then it is also an exclusive subset of $\mathcal{B}$;*
(c) *if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{C}$ are both exclusive subsets of $\mathcal{C}$, then $\mathcal{A} \cup \mathcal{B}$ and $\mathcal{A} \cap \mathcal{B}$ are also exclusive (and hence all exclusive subsets of $\mathcal{C}$ form a lattice).*

The following simple technical lemma dealing with exclusive sets of implicates was proved in [4] (as Lemma 5.4).

Lemma 8 *Let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses (of f) and $\mathcal{C} \subseteq \mathcal{I}(f)$ be a set of clauses such that $\mathcal{X} \subseteq \mathcal{R}(\mathcal{C})$. Then $\mathcal{R}(\mathcal{X}) = \mathcal{R}(\mathcal{C} \cap \mathcal{X})$.*

To see an interesting example of exclusive sets of clauses, let us for a moment return to Horn functions. Let h be a Horn function and let us partition the set $\mathcal{I}(h)$ into two subsets $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ where $\mathcal{H}$ is the set of all pure Horn clauses in $\mathcal{I}(h)$

and $\mathcal{N}$ is the set of all negative clauses in $\mathcal{I}(h)$.[3] Then it is not hard to see that $\mathcal{H}$ is an exclusive set of h (the resolvent is in $\mathcal{H}$ only if both parent clauses are in $\mathcal{H}$).

The partition $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ has some important properties (shown in [15]). The first such property states, that if ϕ_1 and ϕ_2 are two distinct prime CNFs representing h, then the pure Horn parts of ϕ_1 and ϕ_2 (i.e., the conjunctions of all pure Horn clauses in the given CNFs) also represent the same pure Horn function, called in [15] the *pure Horn component* of h.

Proposition 1 [15] *Let ϕ_1 and ϕ_2 be two distinct prime CNFs of a Horn function h. Then $\phi(\mathcal{C}(\phi_1) \cap \mathcal{H}) \equiv \phi(\mathcal{C}(\phi_2) \cap \mathcal{H})$.*

Proposition 1 was generalized in [4] to all exclusive sets (it appears there as Theorem 5.5).

Proposition 2 [4] *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{I}(f)$ be two distinct sets of clauses such that $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{C}_2) \equiv f$, i.e., such that both sets represent f, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Then $\phi(\mathcal{C}_1 \cap \mathcal{X}) \equiv \phi(\mathcal{C}_2 \cap \mathcal{X})$.*

It is immediate to see that Proposition 1 is just a special case of Proposition 2. We can also generalize the notion of a "pure Horn component".

Definition 5 Let f be an arbitrary Boolean function, $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses of f, and $\mathcal{C} \subseteq \mathcal{I}(f)$ be a set of clauses which represents f (i.e., $\phi(\mathcal{C}) \equiv f$). The Boolean function $f_{\mathcal{X}}$ represented by the set $\mathcal{C} \cap \mathcal{X}$ is called the *$\mathcal{X}$-component* of the function f. We shall simply call a function g an *exclusive component of f*, if $g = f_{\mathcal{X}}$ for some exclusive subset $\mathcal{X} \subseteq \mathcal{I}(f)$.

Proposition 2 guarantees that the $\mathcal{X}$-component $f_{\mathcal{X}}$ is well defined for every exclusive set $\mathcal{X} \subseteq \mathcal{I}(f)$. Moreover, Proposition 2 has an important consequence (appearing in [4] as Corollary 5.7) that will prove to be instrumental later in the minimization of CQ functions.

Corollary 3 *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{I}(f)$ be two distinct sets of clauses such that $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{C}_2) \equiv f$, i.e., such that both sets represent f, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Then $\phi((\mathcal{C}_1 \setminus \mathcal{X}) \cup (\mathcal{C}_2 \cap \mathcal{X})) \equiv f$.*

Loosely speaking, Corollary 3 says that if $\mathcal{C}_1, \mathcal{C}_2$ both represent f and $\mathcal{X}$ is exclusive, then removing $\mathcal{C}_1 \cap \mathcal{X}$ from $\mathcal{C}_1$ and replacing it with $\mathcal{C}_2 \cap \mathcal{X}$ produces another representation of f. This statement can be strengthened in the following way: if $\mathcal{C}_1$ is a prime and irredundant representation of f and $\mathcal{C}_2 \cap \mathcal{X}$ is a prime and irredundant representation of $f_{\mathcal{X}}$ then also $(\mathcal{C}_1 \setminus \mathcal{X}) \cup (\mathcal{C}_2 \cap \mathcal{X})$ is a prime and irredundant representation of f. Before stating this stronger version of Corollary 3 we first prove a simple technical lemma.

[3]It is left to the reader to verify the easy fact that $\mathcal{H}$ and $\mathcal{N}$ indeed constitute a partition of $\mathcal{I}(h)$, i.e., that no clause which is neither pure Horn nor negative can appear in $\mathcal{I}(h)$ (recall that each prime implicate of a Horn function is either pure Horn or negative).

Lemma 9 *Let $\mathcal{C} \subseteq \mathcal{I}(f)$ be an irredundant and prime set of clauses such that $\phi(\mathcal{C}) \equiv f$, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Then $\mathcal{C} \cap \mathcal{X}$ is an irredundant and prime set of clauses such that $\phi(\mathcal{C} \cap \mathcal{X}) \equiv f_{\mathcal{X}}$.*

Proof The fact that $\mathcal{C} \cap \mathcal{X}$ represents $f_{\mathcal{X}}$ follows directly from the definition of $f_{\mathcal{X}}$. The irredundancy of $\mathcal{C} \cap \mathcal{X}$ trivially follows from the irredundancy of $\mathcal{C}$ (if we can drop a clause from $\mathcal{C} \cap \mathcal{X}$ without changing the function represented by $\mathcal{C} \cap \mathcal{X}$ then the same clause can be dropped from $\mathcal{C}$ without changing the function represented by $\mathcal{C}$). To show the primality of $\mathcal{C} \cap \mathcal{X}$ let us assume that there exists a nonprime implicate $C' \in \mathcal{C} \cap \mathcal{X}$ of function $f_{\mathcal{X}}$. However, then C' is also a nonprime implicate of f contradicting the primality of $\mathcal{C}$. □

Corollary 4 *Let $\mathcal{C} \subseteq \mathcal{I}(f)$ be an irredundant and prime set of clauses such that $\phi(\mathcal{C}) \equiv f$, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Moreover, let $\mathcal{C}' \subseteq \mathcal{I}(f)$ be an irredundant and prime set of clauses such that $\phi(\mathcal{C}') \equiv f_{\mathcal{X}}$. Then $(\mathcal{C} \setminus \mathcal{X}) \cup \mathcal{C}'$ is an irredundant and prime set of clauses such that $\phi((\mathcal{C} \setminus \mathcal{X}) \cup \mathcal{C}') \equiv f$.*

Proof The fact that $(\mathcal{C} \setminus \mathcal{X}) \cup \mathcal{C}'$ represents f follows directly from Corollary 3 (where $\mathcal{C}'$ plays the role of $\mathcal{C}_2 \cap \mathcal{X}$). The irredundancy and primality of $(\mathcal{C} \setminus \mathcal{X}) \cup \mathcal{C}'$ follows by similar arguments as in the proof of Lemma 9. □

Let us recall next that a subset of the implicates $\mathcal{S} \subseteq \mathcal{I}(f)$ of f is called redundant with respect to f, if $\mathcal{S} \cap \mathcal{C} = \emptyset$ for all irredundant representations $\mathcal{C} \subseteq \mathcal{I}(f)$ of f (i.e., for all minimal sets of implicates for which $\mathcal{R}(\mathcal{C}) = \mathcal{I}(f)$). The following statement was proved in [4] (as Corollary 5.8).

Lemma 10 *For every exclusive set $\mathcal{X} \subseteq \mathcal{I}(f)$ we have $\mathcal{R}(\mathcal{X}) = \mathcal{I}(f_{\mathcal{X}})$, furthermore the set $\mathcal{R}(\mathcal{X}) \setminus \mathcal{X}$ is redundant with respect to $f_{\mathcal{X}}$, as well as with respect to f.*

Let us finally close this section by an important consequence of Corollary 3, namely that in an arbitrary representation $\mathcal{C}$ of f we can replace $\mathcal{C} \cap \mathcal{X}$ by an arbitrary representation of $f_{\mathcal{X}}$ and obtain again a representation of f, whenever $\mathcal{X}$ is an exclusive set. This suggests a decomposition of problem HM, which we summarize in the following statement:

Lemma 11 (Decomposition lemma) *Given a function f, let $\emptyset = \mathcal{X}_0 \subseteq \mathcal{X}_1 \subseteq \cdots \subseteq \mathcal{X}_t$ be a chain of exclusive subsets in which $\mathcal{R}(\mathcal{X}_t) = \mathcal{I}(f)$ and let $\mathcal{C}_i^* \subseteq \mathcal{X}_i \setminus \mathcal{X}_{i-1}$ be minimal subsets such that $\mathcal{R}(\mathcal{X}_{i-1} \cup \mathcal{C}_i^*) = \mathcal{R}(\mathcal{X}_i)$ for $i = 1, ..., t$. Then, $\mathcal{C}^* = \bigcup_{i=1}^t \mathcal{C}_i^*$ is a minimal representation of f (where minimality is with respect to any one of the complexity measures introduced earlier).*

Proof Using Lemma 6 we can show by induction that $\mathcal{R}(\mathcal{C}_1^* \cup \cdots \cup \mathcal{C}_i^*) = \mathcal{R}(\mathcal{R}(\mathcal{C}_1^* \cup \cdots \cup \mathcal{C}_{i-1}^*) \cup \mathcal{C}_i^*) = \mathcal{R}(\mathcal{X}_{i-1} \cup \mathcal{C}_i^*) = \mathcal{R}(\mathcal{X}_i)$, for $i = 1, .., t$, which implies by Lemma 5 that $\bigcup_{j=1}^i \mathcal{C}_j^*$ is a representation of the $\mathcal{X}_i$-component $f_{\mathcal{X}_i}$ of f, for $i = 1, .., t$.

Let us now consider an arbitrary CNF representation $\mathcal{C} \subseteq \mathcal{I}(f)$ of f, and define $\mathcal{C}_i = \mathcal{C} \cap (\mathcal{X}_i \setminus \mathcal{X}_{i-1})$ for $i = 1, ..., t$. By (b) of Lemma 7 we have that $\mathcal{X}_{i-1}$ is an exclusive subset of $\mathcal{X}_i$, for $i = 1, ..., t$. Thus, we can apply Lemma 8 to the function $f_{\mathcal{X}_i}$ and its exclusive subset $\mathcal{X}_{i-1}$, for $i = 1, 2, ..., t$, and obtain inductively by Lemma 6 that $\mathcal{R}(\mathcal{X}_{i-1} \cup \mathcal{C}_i) = \mathcal{R}(\mathcal{X}_i)$, for $i = 1, ..., t$. Therefore, by our choice of $\mathcal{C}_i^*$ we have that the size of $\mathcal{C}_i$ is not smaller than that of $\mathcal{C}_i^*$, for $i = 1, ..., t$ (by the complexity measure we use). Since both considered complexity measures are additive, the statement follows. □

4.2 Essential sets and an orthogonality relation

Let us now introduce the second key concept of this section, which will establish a certain orthogonality relation between sets of implicates and CNF representations.

Definition 6 Given a set $\mathcal{C}$ of clauses, a subset $\mathcal{E} \subseteq \mathcal{C}$ is called an *essential subset of* $\mathcal{C}$ if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{C}$ the following implication holds:

$$R(C_1, C_2) \in \mathcal{E} \Longrightarrow C_1 \in \mathcal{E} \text{ or } C_2 \in \mathcal{E},$$

i.e., the resolvent belongs to $\mathcal{E}$ only if at least one of the parent clauses are from $\mathcal{E}$. In particular, if $\mathcal{C} = \mathcal{I}(f)$ for a Boolean function f, we call $\mathcal{E}$ an essential set of clauses of f (or simply an essential set, if f or $\mathcal{C}$ is clear from the context).

It is easy to see that every exclusive set of clauses (and the set $\mathcal{I}(f)$ in particular) is also essential. We summarize in the following lemma a few simple properties of essential sets. Since all these properties follow directly from Definitions 4 and 6 we shall omit the proofs.

Lemma 12 [4] *Let $\mathcal{C}$ be an arbitrary set of clauses. Then,*

(a) *if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{C}$ are both essential subsets of $\mathcal{C}$, then $\mathcal{A} \cup \mathcal{B}$ is also essential;*
(b) *if $\mathcal{R}(\mathcal{C}) = \mathcal{C}$ and $\mathcal{A}$ is an essential subset of $\mathcal{C}$, then $\mathcal{C} \setminus \mathcal{A}$ is closed under resolution, i.e., $\mathcal{C} \setminus \mathcal{A} = \mathcal{R}(\mathcal{C} \setminus \mathcal{A})$;*
(c) *if $\mathcal{R}(\mathcal{A}) = \mathcal{A}$ and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{B} \setminus \mathcal{A}$ is an essential subset of $\mathcal{C}$;*
(d) *if $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{C}$, $\mathcal{A}$ is an essential subset of $\mathcal{B}$, and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{A}$ is an essential subset of $\mathcal{C}$, as well;*
(e) *if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{C}$, $\mathcal{A} \cap \mathcal{B} \neq \emptyset$, $\mathcal{A}$ is an essential subset of $\mathcal{C}$, and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{A} \cap \mathcal{B}$ is also an essential subset of $\mathcal{C}$.*

To see an interesting example of essential sets, let us consider again a Horn function h and return to the partition of the set $\mathcal{I}(h)$ into two subsets $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ where $\mathcal{H}$ is the set of all pure Horn clauses in $\mathcal{I}(h)$ and $\mathcal{N}$ is the set of all negative clauses in $\mathcal{I}(h)$. Then, it is not hard to see that $\mathcal{N}$ is essential for h (since no two clauses in $\mathcal{N}$ are resolvable, the resolvent is in $\mathcal{N}$ only if *exactly* one of the parent clauses is in $\mathcal{N}$ and the other one is in $\mathcal{H}$).

The orthogonality property of essential sets was proved in [4]. This key proposition (which appears there as Theorem 6.4) shows that every essential set has one (or more) of its clauses present in every representation of f and moreover that this condition is not only necessary but also sufficient.

Proposition 3 [4] *Let $\mathcal{C} \subseteq \mathcal{I}(f)$ be an arbitrary set of clauses. Then $\mathcal{C}$ represents f if and only if $\mathcal{C} \cap \mathcal{E} \neq \emptyset$ for every nonempty essential set of clauses $\mathcal{E} \subseteq \mathcal{I}(f)$.*

Proposition 3 has an obvious corollary: if there exist nonempty essential sets $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_k \subseteq \mathcal{I}(f)$ which are pairwise disjoint, then every representation of f must consist of at least k clauses. Hence, any collection of pairwise disjoint essential sets of clauses provides an easy lower bound on the size (i.e., number of clauses) of a minimal representation of f.

A second important property (proved in [15]) of the partition $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ states that, if ϕ_1 and ϕ_2 are two distinct irredundant CNFs representing h, then ϕ_1 and ϕ_2 both contain the same number of negative clauses.

Proposition 4 [15] *Let ϕ_1 and ϕ_2 be two distinct irredundant CNFs of a Horn function h. Then $|\mathcal{C}(\phi_1) \cap \mathcal{N}| = |\mathcal{C}(\phi_2) \cap \mathcal{N}|$.*

Let us finish this section by recalling a generalization of Proposition 4 (it appears as Theorem 6.12 in [4]), which will come handy in the subsequent sections of this paper.

Proposition 5 [4] *Given a Boolean function f, let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive subset of f such that no two clauses from $\mathcal{E} = \mathcal{I}(f) \setminus \mathcal{R}(\mathcal{X})$ are resolvable. Then, there exists an integer $k = k(\mathcal{E}) > 0$, and pairwise disjoint essential subsets $\mathcal{Q}_j \subseteq \mathcal{E}$, $j = 1, .., k$ such that $|\mathcal{Q}_j \cap \mathcal{C}| = 1$ for $j = 1, ..., k$ and $|(\mathcal{E} \setminus \bigcup_{j=1}^{k} \mathcal{Q}_j) \cap \mathcal{C}| = 0$ for any irredundant set $\mathcal{C} \subseteq \mathcal{I}(f)$ of clauses representing f.*

Of course, it is clear that Proposition 5 implies the following corollary which more closely resembles the statement of Proposition 4.

Corollary 5 [4] *Let f, $\mathcal{X}$, and $\mathcal{E}$ be as in the statement of Proposition 5, and let ϕ_1 and ϕ_2 be two distinct irredundant CNFs of f. Then $|\mathcal{C}(\phi_1) \cap \mathcal{E}| = |\mathcal{C}(\phi_2) \cap \mathcal{E}|$.*

Clearly, Proposition 4 is just a special case of Corollary 5 if we set $\mathcal{X}$ to be the set of all pure Horn clauses in $\mathcal{I}(f)$ (in this case $\mathcal{X} = \mathcal{R}(\mathcal{X})$) and $\mathcal{E}$ to be the set of all negative clauses in $\mathcal{I}(f)$.

5 Horn minimization

Let us return now to the problem of Horn minimization, and let us recall first a few preprocessing steps.

By standard Boolean terminology a *unit* clause is a clause consisting of exactly one literal. If x or $\overline{x}$ is a unit prime implicate of a Boolean function f, then clearly no other prime implicates of f may contain the variable x (negated or not). Therefore any Horn function f represented by a CNF ϕ can be decomposed (in $O(|\phi|_\ell^2)$ time due to Theorem 2) into a conjunction of unit clauses f_1 and a Horn function f_2 which has no unit prime implicates, in such a way that f_1 and f_2 are defined on disjoint sets of variables, and $f = f_1 \wedge f_2$. Since the aim of this paper is Horn minimization, and the above described decomposition of course outputs the shortest possible representation of the exclusive component f_1 of f, we can restrict our attention (without loss of generality) solely to functions with no unit prime implicates. Note that both $\mathcal{I}(f_1)$ and $\mathcal{I}(f_2)$ are exclusive sets, so we can minimize them independently due to Corollary 3.

Moreover, due to Proposition 4 (and its more general form Proposition 5) we may restrict our attention even further to pure Horn functions because the minimization of a Horn function really amounts to the minimization of its pure Horn component. Again note that the set of pure Horn implicates is exclusive and so it can be minimized independently due to Corollary 3.

Therefore, throughout the remainder of this section we can assume that the Horn function h to be minimized is pure Horn and has no unit prime implicates. Let us denote by η the given CNF representation of it.

5.1 Implication graphs

Recall that in Section 3 we have defined the implication graphs $\mathbf{G}_\eta = (\mathbf{N}, \mathbf{A}_\eta)$ and $\mathbf{G}_h = (\mathbf{N}, \mathbf{A}_h)$ associated to η and h, respectively, where $\mathbf{N}$ is the set of variables of the function h represented by the CNF η. We shall first use these implication graphs to define exclusive subsets of clauses for h. In what follows, we shall simply write $\mathbf{G}$ instead of $\mathbf{G}_h$ or $\mathbf{G}_\eta$, whenever h and/or η will be clear from the context.

Definition 7 Given a subset Y of variables of the function h, let us denote by

$$Clauses(Y) = \{C \in \mathcal{I}(h) \mid Vars(C) \subseteq Y\}$$

the set of all clauses from $\mathcal{I}(h)$ which have all their variables belonging to the set Y.

Lemma 13 *Let $X \subseteq \mathbf{N}$ be a set of nodes of the implication graph $\mathbf{G} = \mathbf{G}_h$ (variables of h). Then the sets $Clauses(Cone_{\mathbf{G}}(X))$ and $Clauses(Anticone_{\mathbf{G}}(X))$ are both exclusive sets of clauses of h.*

Proof Let $C_1 = B_1 \vee z$ and $C_2 = B_2 \vee \overline{z} \vee y$ be two resolvable clauses in $\mathcal{I}(h)$ and let $C = R(C_1, C_2) = B_1 \vee B_2 \vee y$ be their resolvent.

If $Vars(C) \subseteq Cone_{\mathbf{G}}(X)$ then $y \in Cone_{\mathbf{G}}(X)$ and $(z, y) \in \mathbf{A}_h$ imply $z \in Cone_{\mathbf{G}}(X)$ and thus $Vars(C_1) \subseteq Cone_{\mathbf{G}}(X)$ and $Vars(C_2) \subseteq Cone_{\mathbf{G}}(X)$ proving that $Clauses(Cone_{\mathbf{G}}(X))$ is exclusive.

Similarly, if $Vars(C) \subseteq Anticone_{\mathbf{G}}(X)$ then the fact that $w \in Anticone_{\mathbf{G}}(X)$ and $(w, z) \in \mathbf{A}$ for every $w \in B_1$ implies $z \in Anticone_{\mathbf{G}}(X)$ and thus $Vars(C_1) \subseteq Anticone_{\mathbf{G}}(X)$ and $Vars(C_2) \subseteq Anticone_{\mathbf{G}}(X)$ proving that $Clauses(Anticone_{\mathbf{G}}(X))$ is exclusive. □

Corollary 6 *Let X and Y be two arbitrary sets of nodes (variables) in $\mathbf{G}$, and let us define $Int_{\mathbf{G}}(X, Y) = Clauses(Anticone_{\mathbf{G}}(X)) \cap Clauses(Cone_{\mathbf{G}}(Y))$ to be the “interval” between X and Y. Then $Int_{\mathbf{G}}(X, Y)$ is an exclusive set of clauses.*

Proof Follows immediately from Lemmas 13 and 7 (exclusiveness is closed under intersection). □

Let us show next a few technical claims about forward chaining and implication graphs.

Lemma 14 (Forward chaining) *Let S be a set of variables, $v \in FC_h(S)$, and let $\mathcal{C} \subseteq \mathcal{I}(h)$ be a minimal set of implicates of h for which $v \in FC_{\mathcal{C}}(S)$. Then, we have*

$$\bigcup_{C \in \mathcal{C}} Vars(C) \subseteq Cone_{\mathbf{G}}(\{v\}).$$

Proof By Lemma 4 we have $Vars(C) \subseteq Cone_{\mathbf{G}}(Head(C))$ for all $C \in \mathcal{I}(h)$. Since $\mathcal{C}$ is a minimal set with the property $v \in FC_{\mathcal{C}}(S)$, we can index its clauses $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ so that $v = Head(C_k)$ and $Head(C_j) \in Subg(C_i)$ for some $i > j$ for all $j = 1, ..., k-1$. This implies hence that $Vars(C_j) \subseteq Cone_{\mathbf{G}}(Head(C_i))$ for some $i > j$ for all $j < k$, from which the statement readily follows, since $v = Head(C_k)$. □

We shall call such a minimal set of clauses $\mathcal{C} \subseteq \mathcal{I}(h)$ for which $v \in FC_{\mathcal{C}}(S)$ a *minimal derivation* of v from S.

Definition 8 Let Y be an arbitrary set of variables and $\mathcal{C} \subseteq \mathcal{I}(h)$ be a set of clauses. Then we shall denote by $\mathcal{C}|_Y$ the set of clauses from $\mathcal{C}$ which have all variables in the set Y, i.e., $\mathcal{C}|_Y = \mathcal{C} \cap Clauses(Y)$. Furthermore, if Y is such that the set $\mathcal{E} = Clauses(Y)$ forms an exclusive set of clauses (of h), we denote by $h|_Y = h_{\mathcal{E}}$ the $\mathcal{E}$-component of h (see Definition 5).

Proposition 2 guarantees that the function $h|_Y$ is well defined whenever the set $Clauses(Y)$ is exclusive. Due to Lemma 13 and Corollary 6 this happens for instance whenever Y is an initial, terminal, or interval set in $\mathbf{G}$.

Lemma 15 (Initial set) *Let I be an initial set in $\mathbf{G}$, and let S be an arbitrary set of variables. Then $FC_h(S) \cap I = FC_{h|_I}(S \cap I)$.*

Proof Let us start by proving the inclusion $FC_h(S) \cap I \subseteq FC_{h|_I}(S \cap I)$. Let us consider an arbitrary variable $v \in FC_h(S) \cap I$, and let $\mathcal{C} \subseteq \mathcal{I}(h)$ be a minimal derivation of v from S. It follows from Lemma 14 that $\bigcup_{C \in \mathcal{C}} Vars(C) \subseteq Cone_{\mathbf{G}}(\{v\})$, and because $v \in I$ and I is an initial set, we also have $Cone_{\mathbf{G}}(\{v\}) \subseteq I$. Therefore, $\bigcup_{C \in \mathcal{C}} Vars(C) \subseteq I$ follows, and hence $v \in FC_{h|_I}(S \cap I)$, which is the desired result.

Now let us prove the opposite inclusion. Since $h|_I$ is defined only on variables from the set I, it follows that $FC_{h|_I}(S \cap I) \subseteq I$. The inclusion $FC_{h|_I}(S \cap I) \subseteq FC_h(S)$ is trivial, and so we get that $FC_{h|_I}(S \cap I) \subseteq FC_h(S) \cap I$. □

5.2 Clause graphs

We shall define yet another directed graph associated to a set of pure Horn clauses and/or to a pure Horn function. As opposed to $\mathbf{G}_h$ and $\mathbf{G}_{\mathcal{C}}$ which are defined on the set of variables, this so called *clause graph* is defined on the set of clauses.

Given a set $\mathcal{C}$ of pure Horn clauses let us define its *clause graph* $\mathbf{D}_{\mathcal{C}} = (\mathbf{V}_{\mathcal{C}}, \mathbf{E}_{\mathcal{C}})$ on the given set of clauses as its vertex set $\mathbf{V}_{\mathcal{C}} = \mathbf{V}(\mathbf{D}_{\mathcal{C}}) = \mathcal{C}$, and where the arc set $\mathbf{E}_{\mathcal{C}} = \mathbf{E}(\mathbf{D}_{\mathcal{C}})$ is defined as follows: For $C_1, C_2 \in \mathcal{C}$ we have $(C_1, C_2) \in \mathbf{E}_{\mathcal{C}}$ if and only if both

(1) $Head(C_1) \in Cone_{\mathbf{G}_{\mathcal{C}}}(Head(C_2))$, and
(2) $Subg(C_1) \subseteq FC_{\mathcal{C}}(Subg(C_2))$.

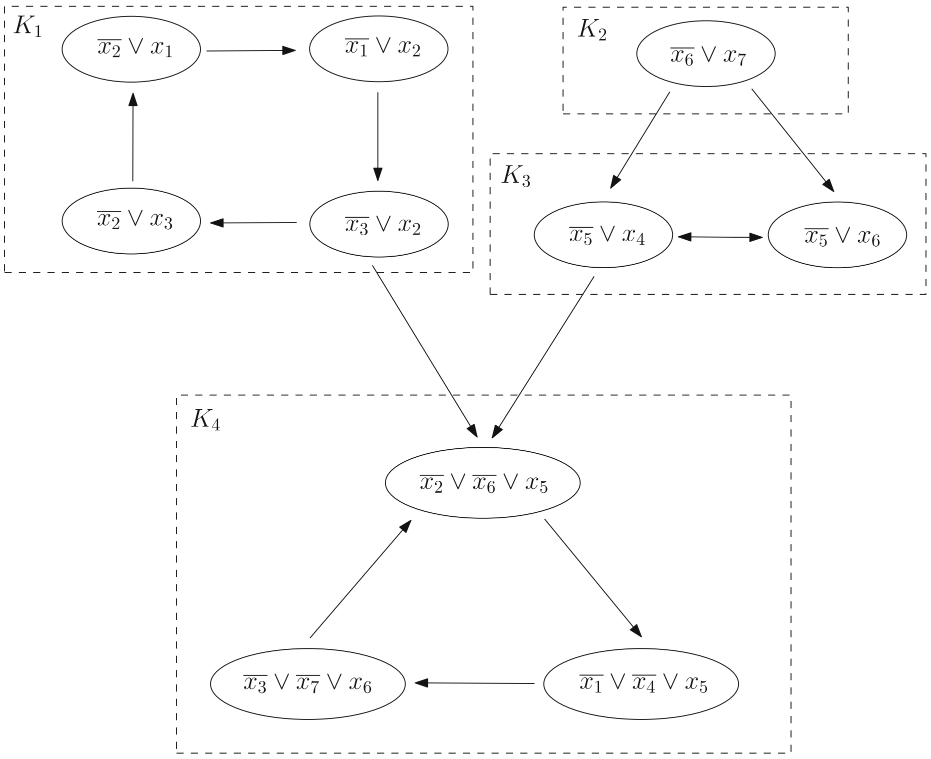
K_1
$\overline{x_2} \vee x_1$
$\overline{x_1} \vee x_2$
$\overline{x_2} \vee x_3$
$\overline{x_3} \vee x_2$
K_2
$\overline{x_6} \vee x_7$
K_3
$\overline{x_5} \vee x_4$
$\overline{x_5} \vee x_6$
K_4
$\overline{x_2} \vee \overline{x_6} \vee x_5$
$\overline{x_3} \vee \overline{x_7} \vee x_6$
$\overline{x_1} \vee \overline{x_4} \vee x_5$

Fig. 3 A clause graph $\mathbf{D}_{\mathcal{C}}$ defined by the set $\mathcal{C}$ of clauses in the CNF ϕ from Example 1

Recall that $FC_{\mathcal{C}}(S)$ is the forward chaining closure of the set of variables S as defined in Section 2.

It is easy to see by the definitions of the implication graph and forward chaining that the clause graph $\mathbf{D}_{\mathcal{C}}$ is transitively closed. In the special case when $\mathcal{C} = \mathcal{I}(h)$ for a pure Horn function h we shall denote the clause graph of $\mathcal{C}$ by $\mathbf{D}_h$, furthermore, whenever the function h will be clear from the context, we shall simply write $\mathbf{D}$ instead of $\mathbf{D}_h$. In this latter case condition (1) simplifies to $(Head(C_1), Head(C_2)) \in \mathbf{A}_h$, due to the fact that $\mathbf{G}_h$ itself is transitively closed.

A clause graph $\mathbf{D}_{\mathcal{C}}$ defined by the set $\mathcal{C}$ of clauses in the CNF ϕ from Example 1 is depicted in Fig. 3. To make the figure easier to read we omitted drawing all the arcs in the transitively closed graph $\mathbf{D}_{\mathcal{C}}$ and displayed its transitive reduction instead. Note that $\mathbf{D}_{\mathcal{C}}$ consists of four strong components denoted by K_1, K_2, K_3 and K_4.

Let us remark that in the sequel, we shall primarily consider clause graphs of Horn functions, though some of those will be specified implicitly via the set of their implicates, such as exclusive components of a given function. For this reason we keep both notations $\mathbf{D}_{\mathcal{I}(h)} = \mathbf{D}_h$.

We shall also need to consider induced subgraphs of clause graphs. For given sets $\mathcal{B} \subseteq \mathcal{C}$ of pure Horn clauses, let us denote by $\mathbf{D}_{\mathcal{C}}(\mathcal{B})$ the subgraph of $\mathbf{D}_{\mathcal{C}}$ induced by the (vertex) set $\mathcal{B}$. Note that by the definitions $\mathbf{D}_{\mathcal{B}}$ is a subgraph of $\mathbf{D}_{\mathcal{C}}$, which may not be induced, i.e., in general we have $\mathbf{D}_{\mathcal{B}} \neq \mathbf{D}_{\mathcal{C}}(\mathcal{B})$. However, it is an induced subgraph when $\mathcal{C} = \mathcal{R}(\mathcal{B})$.

Lemma 16 *Given a set $\mathcal{B}$ of pure Horn clauses, let $\mathcal{C} = \mathcal{R}(\mathcal{B})$. Then, we have $\mathbf{D}_{\mathcal{B}} = \mathbf{D}_{\mathcal{C}}(\mathcal{B})$.*

Proof By the definitions of implication graph and forward chaining we have that $Cone_{\mathbf{G}_{\mathcal{B}}}(v) = Cone_{\mathbf{G}_{\mathcal{C}}}(v)$ for every variable v, and $FC_{\mathcal{B}}(S) = FC_{\mathcal{C}}(S)$ for every set of variables S. □

Let us also remark that while the implication graph $\mathbf{G}_h$ can be built in polynomial time from any given CNF representation $\eta = \phi(\mathcal{C})$, $\mathcal{C} \subseteq \mathcal{I}(h)$ of the function h (according to its definition and Theorem 3, it is enough to construct the transitive closure of the graph $\mathbf{G}_\eta$, which clearly can be done in $O(n^2 + |\eta|_\ell)$ time), it is quite clear that the same goal is impossible to achieve for the clause graph $\mathbf{D}_h$ simply because the set $\mathcal{I}(h)$ may be exponentially large with respect to the size of its CNF representation η. However, induced subgraphs of a clause graph can be built efficiently.

Lemma 17 *Given the pure Horn function h represented by a pure Horn CNF $\eta = \phi(\mathcal{C})$, $\mathcal{C} \subseteq \mathcal{I}(h) = \mathcal{R}(\mathcal{C})$, and two clauses $C_1, C_2 \in \mathcal{I}(h)$, it is possible to verify in linear $O(|\eta|_\ell)$ time whether (C_1, C_2) is an arc of $\mathbf{D}_h$, or not.*

Proof Forward chaining works in linear time in the size of η. Since $Cone_{\mathbf{G}_{\mathcal{C}}}(Subg(C_2)) = Cone_{\mathbf{G}_h}(Subg(C_2))$, condition (1) can also be checked in linear time in the size of η by a direct labeling procedure using only the clauses of $\mathcal{C}$. □

We can save on the above complexity somewhat when building a clause graph, by constructing all arcs entering an implicate essentially at the same price as constructing one of those arcs.

Corollary 7 *For any CNF representation η of the function h and subset $\mathcal{B} \subseteq \mathcal{I}(h)$ the induced subgraph $\mathbf{D}_h(\mathcal{B})$ can be built in $O(|\mathcal{B}|(|\eta|_\ell + |\mathcal{B}|_\ell))$ time. In particular, the induced subgraph $\mathbf{D}_h(\mathcal{C}(\eta))$ can be built in $O(|\eta|_c|\eta|_\ell) = O(m\ell)$ time, where $m = |\eta|_c$ is the number of clauses in η, and $\ell = |\eta|_\ell$ is the number of literals in η.*

Proof Let us build first the implication graph $\mathbf{G}_h$ in $O(|\eta|_\ell)$ time. Next, compute the sets $FC_h(Subg(C))$ for all clauses of $\mathcal{B}$ in $O(|\mathcal{B}||\eta|_\ell)$ time, and store them in a $|\mathcal{B}| \times n$ binary matrix. Finally, for each $C \in \mathcal{B}$ we can check in $O(|C||\mathcal{B}|)$ time which other clauses of $\mathcal{B}$ are reachable from C, thus in another $O(|\mathcal{B}||\mathcal{B}|_\ell)$ time we can construct $\mathbf{D}_h(\mathcal{B})$. □

5.3 Strong components of clause graphs

Let us now derive several important properties of clause graphs.

Lemma 18 *Let $\mathcal{C}$ be a set of Horn clauses, and $C_1, C_2 \in \mathcal{C}$ be two resolvable clauses such that $C = R(C_1, C_2) \in \mathcal{C}$. Then (C_1, C) and (C_2, C) are arcs in $\mathbf{D}_{\mathcal{C}}$.*

Proof Let $C_1 = B_1 \vee v$, $C_2 = B_2 \vee \overline{v} \vee u$, and $C = R(C_1, C_2) = B_1 \vee B_2 \vee u$. Since $C_2 \in \mathcal{C}$ we get that (v, u) is an arc in the implication graph $\mathbf{G}_{\mathcal{C}}$ by definition, and so condition (1) in the definition of $\mathbf{D}_{\mathcal{C}}$ is satisfied for both (C_1, C) and (C_2, C).

To prove condition (2) we have to show that $B_1 \subseteq FC_{\mathcal{C}}(B_1 \cup B_2)$ and $B_2 \cup \{v\} \subseteq FC_{\mathcal{C}}(B_1 \cup B_2)$. Since the inclusions $B_1 \subseteq FC_{\mathcal{C}}(B_1 \cup B_2)$ and $B_2 \subseteq FC_{\mathcal{C}}(B_1 \cup B_2)$ are trivial, we need only to show that $v \in FC_{\mathcal{C}}(B_1 \cup B_2)$, which follows by the fact that $C_1 = B_1 \vee v$ is a clause of $\mathcal{C}$ and hence $v \in FC_{\mathcal{C}}(B_1) \subseteq FC_{\mathcal{C}}(B_1 \cup B_2)$. □

Theorem 6 *Let $\mathcal{C}$ be a set of Horn clauses, and $\mathcal{I} \subseteq \mathcal{C}$ be an initial set in $\mathbf{D}_{\mathcal{C}}$. Then $\mathcal{I}$ is an exclusive subset of $\mathcal{C}$.*

Proof Let $C_1, C_2 \in \mathcal{C}$ be two resolvable clauses such that $C = R(C_1, C_2) \in \mathcal{C}$. Then by Lemma 18 $(C_1, C) \in \mathbf{E}_{\mathcal{C}}$ and $(C_2, C) \in \mathbf{E}_{\mathcal{C}}$ and hence both $C_1 \in \mathcal{I}$ and $C_2 \in \mathcal{I}$, because $\mathcal{I}$ is an initial set in $\mathbf{D}_{\mathcal{C}}$. □

Let us show next some important properties of the strong components of the clause graph.

Theorem 7 *Let h be a pure Horn function, and $\mathcal{K}$ be a strong component of $\mathbf{D}_h$. Then, either $\mathcal{K}$ is redundant or $\mathcal{K}$ contains a nonempty essential set of clauses.*

Proof Let us note first that since $\mathcal{K}$ is a strong component of $\mathbf{D} = \mathbf{D}_h$, both $Cone_{\mathbf{D}}(\mathcal{K})$ and $\mathcal{X} = Cone_{\mathbf{D}}(\mathcal{K}) \setminus \mathcal{K}$ are initial sets of $\mathbf{D}$, and hence by Theorem 6 both of them are exclusive (for h). Furthermore, by Lemma 10 the set $\mathcal{S} = \mathcal{R}(\mathcal{X}) \setminus \mathcal{X}$ is redundant. Thus, if $\mathcal{K} \subseteq \mathcal{S}$, then $\mathcal{K}$ itself is redundant. On the other hand, if $\mathcal{K} \not\subseteq \mathcal{S}$, then the set $\mathcal{E} = \mathcal{K} \setminus \mathcal{R}(\mathcal{X}) = Cone_{\mathbf{D}}(\mathcal{K}) \setminus \mathcal{R}(\mathcal{X}) \subseteq \mathcal{K}$ is essential by (c) of Lemma 12. □

An important consequence of the above theorem is that the number of non-redundant strong components of $\mathbf{D}$ is limited by the size of an arbitrary representation of h.

Corollary 8 *Given an arbitrary irredundant representation $\mathcal{C} \subseteq \mathcal{I}(h)$ of h, the strong components of the graph $\mathbf{D}_{\mathcal{C}} = \mathbf{D}(\mathcal{C})$ are in a one-to-one correspondence with the non-redundant strong components of $\mathbf{D}$.*

Proof For any strong component $\mathcal{K}$ of $\mathbf{D}$, the set $\mathcal{K} \cap \mathcal{C}$ forms a strong component of $\mathbf{D}(\mathcal{C})$, by definition of an induced subgraph. Since $\mathcal{R}(\mathcal{C}) = \mathcal{I}(h)$, we have the equality $\mathbf{D}_{\mathcal{C}} = \mathbf{D}(\mathcal{C})$ by Lemma 16. Thus, by Proposition 3 and Theorem 7, non-redundant strong components of $\mathbf{D}$ correspond in a one to one way to the (nonempty) strong components of $\mathbf{D}_{\mathcal{C}}$. □

Now we are ready to describe an efficient procedure to construct a chain of exclusive sets, in the spirit of Lemma 11.

Theorem 8 *Let h be a pure Horn function on n variables with no unit implicates, represented by an irredundant and prime CNF $\eta = \phi(\mathcal{C})$, $\mathcal{C} \subseteq \mathcal{I}(h)$. Let $K_1, \ldots, K_t$ be* *all strong components of graph $D_{\mathcal{C}}$ sorted according to some topological order. Let us define sets of clauses $\mathcal{X}_0 = \emptyset$ and*

$$\mathcal{X}_i = \bigcup_{j=1}^{i} Cone_{\mathbf{D}_h}(K_j),$$

then the following properties are satisfied:

(i) *$\mathcal{X}_{i-1}$ is an exclusive subset of $\mathcal{X}_i$ for $i = 1, \dots, t$;*
(ii) *$\mathcal{X}_t$ is an exclusive subset of $\mathcal{I}(h)$ and $\mathcal{R}(\mathcal{X}_t) = \mathcal{I}(f)$;*
(iii) *$K_i = \mathcal{C} \cap (\mathcal{X}_i \setminus \mathcal{X}_{i-1})$ and the set K_i is the unique terminal strong component of $\mathbf{D}_{Cone_{\mathbf{D}_{\mathcal{X}_i \cap \mathcal{C}}}(K_i)}$, for $i = 1, ..., t$;*
(iv) *the strong components $K_1, \dots, K_t$ can be found in time $O(n^2 + m\ell)$, where m is the number of clauses of η and ℓ is the number of literals of η.*

Proof Since $\mathcal{X}_{i-1}$ is by definition an initial set of graph $\mathbf{D}_h$ for every $i = 1, \dots, t$, it is an exclusive subset of $\mathcal{I}(h)$ according to Theorem 6. Therefore according to Lemma 7, proposition (b), it is also an exclusive subset of $\mathcal{X}_i$. Therefore property (i) is satisfied.

Since $\mathcal{C} \subseteq \mathcal{X}_t$ and $\eta = \phi(\mathcal{C})$ is a prime representation of h, clearly $\mathcal{R}(\mathcal{X}_t) = \mathcal{I}(h)$. Because $\mathcal{X}_t$ is an initial set of graph D_h, it is also an exclusive subset of $\mathcal{I}(h)$ due to Theorem 6. Hence (ii) is satisfied.

The fact that $K_i = \mathcal{C} \cap (\mathcal{X}_i \setminus \mathcal{X}_{i-1})$ follows directly from the definition of sets $\mathcal{X}_i$ and $\mathcal{X}_{i-1}$. Set K_i is clearly the unique strong component of $D_{Cone_{\mathbf{D}_{\mathcal{X}_i \cap \mathcal{C}}}(K_i)}$ by the definition of $Cone_{\mathbf{D}_{\mathcal{X}_i \cap \mathcal{C}}}(K_i)$. Hence also property (iii) is satisfied.

Now, let us examine, how quickly we can find strong components $K_1, \dots, K_t$. We first construct an implication graph $\mathbf{G}_{\mathcal{C}}$, which can be done in $O(\ell)$ time and then its transitive closure $\mathbf{G}_h$ can be found in $O(n^2)$ time. Therefore $O(n^2 + \ell)$ time suffices to perform this step.

Then the subgraph $\mathbf{D}_{\mathcal{C}}$ of the clause graph $\mathbf{D}_h$ is constructed. For this purpose it is necessary to test for each pair of clauses $C_1, C_2 \in \mathcal{C}$, whether (C_1, C_2) is an arc in $\mathbf{D}_{\mathcal{C}}$, which amounts to verifying the conditions:

1. $(Head(C_1), Head(C_2))$ is an arc in $\mathbf{G}_h$ or $Head(C_1) = Head(C_2)$). This can be tested in constant time because $\mathbf{G}_h$ is already constructed (assume for simplicity that the first step constructs its adjacency matrix).
2. $Subg(C_1) \subseteq FC_h(Subg(C_2))$). In order to test this condition we shall first construct for each clause C the set $FC_h(Subg(C))$. By Lemma 3, this takes $O(\ell)$ time per clause and thus $O(m\ell)$ time in total (the results can be stored e.g. in an $m \times n$ matrix). Now we can easily decide whether $Subg(C_1) \subseteq FC_h(Subg(C_2))$ in $O(|C_1|)$ time, and thus we can find all arcs leading to the clause C_2 in $O(\ell)$ time.

Therefore all arcs of graph $D_{\mathcal{C}}$ can be discovered in $O(m\ell)$ time. Strong components and their topological order can be found in time linear in the size of $D_{\mathcal{C}}$, i.e., in $O(m^2) = O(m\ell)$ time. By this also the property (iv) is satisfied. □

An important consequence of the above theorem is that the problem of Horn Minimization can be reduced to the following special incremental problem, of finding the optimal representation of a terminal strong component of the clause graph (or in short, the problem of ORTSC).

PROBLEM ORTSC($\mathcal{F}$, $\mathcal{Q}$)

Input: A pure Horn function g, represented by a prime and irredundant set of clauses $\mathcal{F} \cup \mathcal{Q}$, for which $\mathcal{F} \subseteq \mathcal{X}$ for some exclusive subset $\mathcal{X}$ of $\mathcal{I}(g)$ and $\mathcal{Q} \subseteq \mathcal{K} = \mathcal{I}(g) \setminus \mathcal{X}$, and where $\mathcal{K}$ is the unique terminal strong component of the clause graph $\mathbf{D}_g$ (in other words, $\mathcal{I}(g) = \mathcal{X} \cup \mathcal{K} = Cone_{\mathbf{D}_g}(\mathcal{K})$).

Output: A minimum cardinality subset $\mathcal{Q}^* \subseteq \mathcal{K}$, such that $\mathcal{R}(\mathcal{F} \cup \mathcal{Q}^*) = \mathcal{R}(\mathcal{F} \cup \mathcal{Q}) = \mathcal{I}(g)$.

Theorem 9 *If problem ORTSC can be solved in polynomial time, then problem HM can be solved in polynomial time.*

Proof Given a Horn function represented by a Horn CNF, let us first bring it to a prime and irredundant form. We can then delete the negative clauses, according to Propositions 1 and 4. Let us next perform the preprocessing steps as described in the beginning of Section 5. All these can be done in $O(\ell^2)$ time, and reduces the minimization of the original input to the minimization of a pure Horn function h which has no unit prime implicates, and which is represented by an irredundant and prime set $\mathcal{C}$ of clauses.

Let us next find strong components $K_1, \ldots, K_t$ of graph $\mathbf{D}_\mathcal{C}$ and their topological order in time $O(n^2 + m\ell)$ as in Theorem 8, and let $\mathcal{X}_1, \ldots, \mathcal{X}_t$ be defined as in Theorem 8.

For every $i = 1, \ldots, t$ the set $\mathcal{X}_i \setminus \mathcal{X}_{i-1}$ contains the (non-redundant) strong component L_i of the graph $\mathcal{D}_h$ which corresponds by Corollary 8 to the strong component K_i of the graph $\mathbf{D}_\mathcal{C}$. In other words $K_i = \mathcal{C} \cap L_i \subseteq L_i \subseteq \mathcal{X}_i \setminus \mathcal{X}_{i-1}$ holds (where the last inclusion may be proper, if $\mathcal{X}_i \setminus \mathcal{X}_{i-1}$ contains some redundant strong components of $\mathbf{D}_h$). Therefore, in order to satisfy the assumptions of Lemma 11 (which yields a minimum cardinality set of clauses which represents h and thus solves HM) it suffices to show, that by solving an instance of ORTSC with suitable input we can find for every $i = 1, \ldots, t$ a subset $K_i^* \subseteq \mathcal{X}_i \setminus \mathcal{X}_{i-1}$ such that

$$K_i^* \text{ is a minimum cardinality set such that } \mathcal{R}(\mathcal{X}_{i-1} \cup K_i^*) = \mathcal{R}(\mathcal{X}_i). \tag{3}$$

The set $\mathcal{X}_i$ can be split into two disjoint parts: $\mathcal{V} = Cone_{\mathbf{D}_h}(K_i)$ and $\mathcal{W} = \mathcal{X}_i \setminus \mathcal{V}$. Let us denote $\mathcal{Y} = \mathcal{V} \cap \mathcal{X}_{i-1}$. Note that both $\mathcal{V}$ and $\mathcal{Y}$ are again initial sets of $\mathbf{D}_h$, so both are exclusive subsets of $\mathcal{I}(h)$ by Theorem 6. Moreover $\mathcal{W} = \mathcal{X}_{i-1} \setminus \mathcal{Y}$ holds and so $\mathcal{X}_{i-1}$ can be rewritten as disjoint union of $\mathcal{W}$ and $\mathcal{Y}$. Thus (3) can be rewritten as

$$K_i^* \text{ is a minimum cardinality set such that } \mathcal{R}(\mathcal{W} \cup \mathcal{Y} \cup K_i^*) = \mathcal{R}(\mathcal{W} \cup \mathcal{V}). \tag{4}$$

Let us show that condition (3) is equivalent to the condition

$$K_i^* \text{ is a minimum cardinality set such that } \mathcal{R}(\mathcal{Y} \cup K_i^*) = \mathcal{R}(\mathcal{V}) \tag{5}$$

To show that (4) implies (5) it suffices to use Lemma 8 where we set $\mathcal{X} = \mathcal{V}$ and $\mathcal{C} = \mathcal{W} \cup \mathcal{Y} \cup K_i^*$ and $\mathcal{C} = \mathcal{W} \cup \mathcal{Y} \cup K_i^*$. For the reverse implication we need to use Lemma 6 twice. First we set $\mathcal{X} = \mathcal{W}$ and $\mathcal{Z} = \mathcal{Y} \cup K_i^*$ obtaining $\mathcal{R}(\mathcal{W} \cup (\mathcal{Y} \cup K_i^*)) = \mathcal{R}(\mathcal{W} \cup \mathcal{R}(\mathcal{Y} \cup K_i)) = \mathcal{R}(\mathcal{W} \cup \mathcal{R}(\mathcal{V}))$. In the next step we set $\mathcal{X} = \mathcal{W}$ and $\mathcal{Z} = \mathcal{V}$ getting $\mathcal{R}(\mathcal{W} \cup \mathcal{R}(\mathcal{V})) = \mathcal{R}(\mathcal{W} \cup \mathcal{V})$.

A set K_i^* satisfying condition (5) can be found by solving ORTSC for the instance in which we set g to be the $\mathcal{Y}$-component of h, $\mathcal{F} = \mathcal{C} \cap \mathcal{Y} = Cone_{D_\mathcal{C}}(K_i) \setminus K_i$, and $\mathcal{Q} = \mathcal{C} \cap (\mathcal{V} \setminus \mathcal{Y}) = K_i$. According to (i)–(iii) of Theorem 8, the function g represented by $\mathcal{F} \cup \mathcal{Q} = \mathcal{C} \cap \mathcal{X}_i$ satisfies the conditions of an input for ORTSC. Thus the set $K_i^* = \mathcal{Q}^*$ which is a solution of ORTSC to this input satisfies condition (5) and therefore by the above arguments also condition (3). Hence the sets $K_1^*, \ldots, K_t^*$ satisfy requirements of Lemma 11, and $\mathcal{C}^* = \bigcup_{i=1}^t K_i^*$ is a minimum cardinality set representing h. □

Remark 1 If ORTSC is reformulated to output a set $\mathcal{Q}^*$ consisting of a minimum number of literals such that $\mathcal{R}(\mathcal{F} \cup \mathcal{Q}^*) = \mathcal{R}(\mathcal{F} \cup \mathcal{Q})$, then the proof of Theorem 9 can be easily modified to show that using ORTSC one can find a representation of a given input pure Horn function with the minimum number of literals. In this case the minimization procedure does not necessarily work for Horn CNFs which contain negative clauses. The same number of negative clauses may have a different total number of literals, so dropping them in the preprocessing step cannot be justified in this case (see Section 6.2 for more details on the minimization of the number of literals).

5.4 Switching lemmas

To arrive at a polynomial algorithm for Horn minimization (at least for CQ-functions) we need to tackle problem ORTSC, according to Theorem 9. For this, we shall need to analyze further the structure of clause graphs and in particular, the structure of clause graphs of CQ-functions.

Let us remark that until this point, all of our statements remained valid for arbitrary additive complexity measure of CNF-s, in particular, for both measures $|\eta|_\ell$ and $|\eta|_c$. From now on however, we need to restrict our claims to the minimality of the number of clauses, i.e., to measure $|\eta|_c$.

In this subsection, we prove a series of "switching" claims for pure Horn, and more specifically for CQ functions. In each of these claims some literals of implicates connected in the clause graph are switched, resulting in another implicate of h. We assume h to be a fixed pure Horn function, and simply write $\mathbf{D}$ instead of $\mathbf{D}_h$, and $\mathbf{G}$ instead of $\mathbf{G}_h$.

Lemma 19 *Let $A \vee u$, $B \vee v \in \mathcal{I}(h)$ be implicates of h such that $(B \vee v, A \vee u)$ is an arc of the clause graph* $\mathbf{D}$. *Then, there exists a subset $A' \subseteq A$ such that $A' \vee v$ is a prime implicate of h.*

Proof Since $(B \vee v, A \vee u)$ is an arc of $\mathbf{D}$, we have $B \subseteq FC_h(A)$ by definition, and thus $v \in FC_h(A)$ follows, since $B \vee v \in \mathcal{I}(h)$ is assumed. Therefore, $A \vee v$ is an implicate of h by Lemma 2. Consequently, there must exist a pure Horn prime implicate $A' \vee v$ subsuming $A \vee v$. □

It follows from the definition of the clause graph $\mathbf{D}$ that whenever both (C_1, C_2) and (C_2, C_1) are arcs in $\mathbf{D}$ for some pair of clauses $C_1, C_2 \in \mathcal{I}(h)$, then both $(Head(C_1), Head(C_2))$ and $(Head(C_2), Head(C_1))$ are arcs in the implication graph $\mathbf{G}$. Therefore, all clauses from the same strong component of $\mathbf{D}$ must have their heads in a single strong component of $\mathbf{G}$ (however, heads of clauses from several strong components of $\mathbf{D}$ may also belong to the same strong component of $\mathbf{G}$). This fact allows us to state the following definition.

Definition 9 For a strong component $\mathcal{K}$ of the clause graph $\mathbf{D}$ let us denote by $Q(\mathcal{K})$ the strong component of the implication graph $\mathbf{G}$ which contains the heads of the clauses belonging to $\mathcal{K}$.

In the rest of this section we shall separate, for each implicate $C \in \mathcal{I}(h)$, the subgoals of C which are in the same strong component of the implication graph $\mathbf{G}$ as its head $Head(C)$ from those subgoals of C which are in preceding strong components. This separation will allow us to state and prove several stronger "switching" results. Let us start with a simple auxiliary lemma.

Lemma 20 *Let $\mathcal{K}$ be a strong component of graph $\mathbf{D}$ and let $(A \vee X \vee u)$, $(B \vee Y \vee v) \in \mathcal{K}$ be two implicates of h, such that $A \cap Q(\mathcal{K}) = \emptyset$, $B \cap Q(\mathcal{K}) = \emptyset$, and $X, Y \subseteq Q(\mathcal{K})$. Let further $I = Cone_{\mathbf{G}}(Q(\mathcal{K})) \setminus Q(\mathcal{K})$ be an initial set of $\mathbf{G}$, and denote, as before, by h_I the $Clauses(I)$-component of h, and let $\mathcal{X} = Cone_{\mathbf{D}}(\mathcal{K}) \setminus \mathcal{K}$. Then, we have*

$$B \subseteq FC_{h_I}(A) \cap FC_{\mathcal{X}}(A) \subseteq FC_h(A) \quad \text{and} \quad A \subseteq FC_{h_I}(B) \cap FC_{\mathcal{X}}(B) \subseteq FC_h(B).$$

Proof Let us first recall that h_I is well defined by Definition 8 (applied to the initial set I of the implication graph $\mathbf{G}$). Let us note next that, since $\mathcal{K}$ is a strong component, $(B \vee Y \vee v, A \vee X \vee u)$ is an arc in $\mathbf{D}$, and thus the inclusion $B \cup Y \subseteq FC_h(A \cup X)$ holds (by the definition of the clause graph $\mathbf{D}$). By Lemma 15 applied to the initial set I and the set of variables $A \cup X$ we get

$$B = (B \cup Y) \cap I \subseteq FC_h(A \cup X) \cap I = FC_{h|_I}((A \cup X) \cap I) = FC_{h|_I}(A) \subseteq FC_h(A).$$

Consider next an arbitrary variable $b \in B \setminus A$, and choose a minimal set $\mathcal{C}$ of prime implicates of h_I such that $b \in FC_{\mathcal{C}}(A)$ holds. We claim that $\mathcal{C} \subseteq \mathcal{X}$. To see this claim, let us note first that $\bigcup_{C \in \mathcal{C}} Vars(C) \subseteq Cone_{\mathbf{G}}(b)$ follows by Lemma 14, implying that $(Head(C), b)$ is an arc in $\mathbf{G}$ for every clause $C \in \mathcal{C}$. We also have (b, v) as an arc in $\mathbf{G}$ for all $b \in B$ by Lemma 4, since $B \vee Y \vee v$ is an implicate of h. Finally, we have (v, u) as an arc of $\mathbf{G}$, since $A \vee X \vee u$ and $B \vee Y \vee v$ are implicates from the same strong component $\mathcal{K}$ of $\mathbf{D}$. Since $\mathbf{G}$ is transitively closed, these imply that $(Head(C), u)$ is an arc in $\mathbf{G}$ for all $C \in \mathcal{C}$. On the other hand, for all $C \in \mathcal{C}$ we have $Subg(C) \subseteq FC_{\mathcal{C}}(A) \subseteq FC_h(A) \subseteq FC_h(A \cup X)$ by our choice of $\mathcal{C}$. Thus, $(C, A \vee X \vee u) \in \mathbf{A}_h$ follows, implying $\mathcal{C} \subseteq Cone_{\mathbf{D}}(\mathcal{K})$. Since $\mathcal{C} \subseteq Clauses(I)$ and $Clauses(I) \cap \mathcal{K} = \emptyset$, $\mathcal{C} \subseteq \mathcal{X}$ follows, as claimed.

Applying the above claim for all $b \in B \setminus A$, we get $B \subseteq FC_{\mathcal{X}}(A) \subseteq FC_h(A)$, completing the proof of the first part of the theorem.

The second part, i.e., the inclusion $A \subseteq FC_{h_I}(B) \cap FC_{\mathcal{X}}(B) \subseteq FC_h(B)$ can be proved analogously by interchanging the roles of clauses $A \vee X \vee u$ and $B \vee Y \vee v$. $\square$

Now we are ready to state the main result of this subsection, a generalization of Lemma 19.

Theorem 10 (Switching Theorem) *Let $\mathcal{K}$ be a strong component of the clause graph $\mathbf{D} = \mathbf{D}_h$ and let $A \vee X \vee u,\ B \vee Y \vee v \in \mathcal{K}$ be two implicates of h, such that $A \cap Q(\mathcal{K}) = \emptyset$, $B \cap Q(\mathcal{K}) = \emptyset$, and $X, Y \subseteq Q(\mathcal{K})$. Then, there exist subsets $A' \subseteq A$ and $Y' \subseteq Y$, such that*

(a) *$A' \vee Y' \vee v$ is a prime implicate of h belonging to $Cone_{\mathbf{D}}(\mathcal{K})$.*
(b) *Additionally, if $B \vee Y \vee v$ is a prime implicate of h and $Y \neq \emptyset$, then also $Y' \neq \emptyset$.*
(c) *Furthermore, if $B \vee Y \vee v$ is not redundant, then $A' \vee Y' \vee v \in \mathcal{K}$.*
(d) *Finally, if $A \vee X \vee u$ is prime, and $A' \vee Y' \vee v \in \mathcal{K}$, then $A = A'$.*

Proof By Lemma 20 we get $B \subseteq FC_h(A)$. Moreover, $B \vee Y \vee v$ is an implicate of h, and so it follows that $v \in FC_h(A \cup Y)$, which by Lemma 2 implies that $A \vee Y \vee v$ is an implicate of h. Thus there must exist sets $A' \subseteq A$ and $Y' \subseteq Y$, such that $A' \vee Y' \vee v \in \mathcal{I}^p(h)$. Furthermore, since $(B \vee Y \vee v,\ A \vee X \vee u)$ is an arc in the clause graph $\mathbf{D}$, (v, u) must be an arc of $\mathbf{G}$ and $A' \cup Y' \subseteq A \cup Y \subseteq A \cup B \cup Y \subseteq FC_h(A \cup X)$ must hold, implying that $(A' \vee Y' \vee v,\ A \vee X \vee u)$ is an arc in the implication graph $\mathbf{D}$, completing the proof of (a).

To see (b), let us assume that $B \vee Y \vee v \in \mathcal{I}^p(h)$, $Y \neq \emptyset$, and by contradiction that $Y' = \emptyset$. These imply that $A' \vee v \in \mathcal{I}^p(h)$ and so $v \in FC_h(A')$. By Lemma 20 we get $A' \subseteq A \subseteq FC_h(B)$, implying $v \in FC_h(B)$, from which $B \vee v \in \mathcal{I}(h)$ would follow, contradicting the primality of $B \vee Y \vee v$.

Next, to see (c), let us define $\mathcal{X} = Cone_{\mathbf{D}}(\mathcal{K}) \setminus \mathcal{K}$, and observe that $A \subseteq FC_{\mathcal{X}}(B)$ follows by Lemma 20, implying $A' \cup Y' \subseteq A \cup Y \subseteq FC_{\mathcal{X}}(B \cup Y)$. Now, if $A' \vee Y' \vee v$ were not belonging to $\mathcal{K}$, then by (a) it would belong to $\mathcal{X}$, and thus $v \in FC_{\mathcal{X}}(B \cup Y)$ would also follow, implying $B \vee Y \vee v \in \mathcal{R}(\mathcal{X}) \setminus \mathcal{X}$, that is that $B \vee Y \vee v$ is redundant by Lemma 10, since $\mathcal{X}$ is an exclusive set.

Finally, to see (d), let us apply (a) for the pair $A' \vee Y' \vee v, A \vee X \vee u \in \mathcal{K}$ of clauses, and get that $A'' \vee X'' \vee u$ is also an implicate of h for some $X'' \subseteq X$ and $A'' \subseteq A' \subseteq A$. Since $A \vee X \vee u$ is assumed to be prime, $A'' = A' = A$ and $X'' = X$ are implied. □

Corollary 9 *Let $\mathcal{C}$ be a prime and irredundant representation of the pure Horn function h, $\mathcal{K}$ be a strong component of $\mathbf{D}$, and $A \vee X \vee u,\ B \vee Y \vee v \in \mathcal{C} \cap \mathcal{K}$. Then, there exists a subset $Y' \subseteq Y$ such that the set $\mathcal{C}' = (\mathcal{C} \setminus \{B \vee Y \vee v\}) \cup \{A \vee Y' \vee v\}$ is again a prime and irredundant representation of h. Furthermore, if $|Y| \leq 1$, then we must have $Y' = Y$.*

Proof By (a), (c) and (d) of Theorem 10 we know that there exists a subset $Y' \subseteq Y$ such that $A \vee Y' \vee v \in \mathcal{K}$ is a prime implicate of h. Thus both the primality and irredundancy of $\mathcal{C}'$ is implied trivially by that of $\mathcal{C}$. The only claim remained to show is that $\mathcal{C}'$ represents h, that is that $B \vee Y \vee v \in \mathcal{R}(\mathcal{C}')$, or equivalently by Lemma 2 that $v \in FC_{\mathcal{C}'}(B \cup Y)$.

Let us denote by $\mathcal{X} = Cone_{\mathbf{D}}(\mathcal{K}) \setminus \mathcal{K}$ and note that by Lemma 20 we have $A \subseteq FC_{\mathcal{X}}(B)$. Thus, by $\mathcal{C} \cap \mathcal{X} = \mathcal{C}' \cap \mathcal{X}$, we get $A \subseteq FC_{\mathcal{C}'}(B)$, from which $A \cup Y' \subseteq A \cup Y \subseteq FC_{\mathcal{C}'}(B \cup Y)$ follows. Since we also have $A \vee Y' \vee v \in \mathcal{C}'$, $v \in FC_{\mathcal{C}'}(B \cup Y)$ follows too, as claimed.

Finally, by (b) of Theorem 10, we have $Y' \neq \emptyset$ whenever $Y \neq \emptyset$, which implies $Y' = Y$, since $|Y| \leq 1$ is assumed. □

5.5 A classification of strong components of clause graphs

In this section we introduce a useful classification of the strong components of the clause graph **D** by the number of their subgoal variables which belong to the same strong component of the implication graph as their head. Let us start with an analogous classification of the clauses.

Definition 10 A pure Horn clause $C \in \mathcal{I}(h)$ is said to be of type i (with respect to h), if exactly i subgoals of C belong to the same strong component of the graph **G** as the head of C.

Let us note next an easy consequence of Lemma 20.

Corollary 10 *Let $\mathcal{K}$ be a strong component of the clause graph* **D** *and let $(A \vee X \vee u)$, $(B \vee Y \vee v) \in \mathcal{K}$ be two prime implicates of h, such that $A \cap Q(\mathcal{K}) = \emptyset$, $B \cap Q(\mathcal{K}) = \emptyset$, and $X, Y \subseteq Q(\mathcal{K})$. Then, if $X = \emptyset$ we must also have $Y = \emptyset$.*

Proof Since $(B \vee Y \vee v, A \vee X \vee u)$ is an arc of **D**, we have $B \cup Y \subseteq FC_h(A \cup X)$ by the definition of the arcs in the clause graph. By Lemma 20 we also have $A \subseteq FC_h(B)$. Thus, if $X = \emptyset$, then $Y \subseteq FC_h(B)$ follows, implying by Lemma 2 that $B \vee v$ is also an implicate of h. Thus, $Y = \emptyset$ is implied by the primality of $B \vee Y \vee v$. □

Corollary 11 *Let h be a pure Horn function, and let $\mathcal{K}$ be a strong component of* **D**. *Then, either all or none of the prime implicates belonging to $\mathcal{K}$ are of type* 0.

Proof Immediate by Corollary 10. □

If h is a CQ function then Corollary 11 justifies a complete classification of strong components of **D**. Indeed, every prime implicate of h is either of type 0 or of type 1 and hence every strong component of **D** either contains only prime implicates of type 0 or it contains only prime implicates of type 1.

Definition 11 Let $\mathcal{K}$ be a strong component of the graph **D**. Then we say that $\mathcal{K}$ is of type 0 if all prime implicates belonging to $\mathcal{K}$ are of type 0, and we say that $\mathcal{K}$ is of type 1 if all prime implicates belonging to $\mathcal{K}$ are of type 1.

6 Algorithm for minimizing CQ functions

Now we are ready to design an algorithm for the minimization of CQ functions.

Algorithm 1 Minimization of CQ functions

Input: Irredundant and prime CNF $\mathcal{C}$ representing a function h from the class CQ.
Output: A CNF $\mathcal{C}_{min}$ representing h such that $\mathcal{C}_{min}$ has the minimum possible number of clauses.

1. **[Implication graph]** Construct the graph $\mathbf{G}_{\mathcal{C}}$, its transitive closure $\mathbf{G}_h$ and find its strong components.
2. **[Clause graph]** Construct the graph $\mathbf{D}_{\mathcal{C}}$ (i.e., the subgraph of $\mathbf{D}_h$ induced by the clauses in $\mathcal{C}$), find its strong components, and find some topological order $T = (K_1, \ldots K_t)$ of these strong components.
3. **[Main loop]** Process the strong components of the graph $\mathbf{D}_{\mathcal{C}}$ in the topological order T found in the preceding step, and for each component K perform one of the following actions:

   – **Action $\mathcal{K}_0$:** If the component K is of type $\mathcal{K}_0$, then do nothing.
   – **Action $\mathcal{K}_1$:** If the component K is of type $\mathcal{K}_1$, then do the following:

     (a) Switch all subgoal sets outside of $Q(K)$ of all clauses in K to some representative set $A(K)$ (i.e., to a set of subgoals outside of $Q(K)$ of an arbitrary clause from K) obtaining set of clauses K' and new representation $\mathcal{C}'$.
     (b) Construct the subgraph $Q'(K')$ of $Q(K')$ whose arcs are generated only by the clauses from $Cone_{\mathbf{D}_{\mathcal{C}'}}(K')$ and mark all arcs generated by clauses from $(Cone_{\mathbf{D}_{\mathcal{C}'}}(K') \setminus K')$ as *fixed* and all arcs generated by clauses from K' as *free*.
     (c) Find the transitive reduction of $Q'(K')$ with the added restriction that the reduction must contain all fixed arcs. Free arcs can be removed and replaced by other arcs (however the transitive closure of $Q'(K')$ must of course remain the same). This task can be performed using a generalization of the augmentation algorithm described in [12] and [28].
     (d) For each removed arc, remove from $\mathcal{C}'$ the corresponding clause. For each added arc, add to $\mathcal{C}'$ the corresponding quadratic clause and extend the clause by adding the representative set $A(K)$ of subgoals. Update the graph $\mathbf{D}_{\mathcal{C}'}$ accordingly.

4. **[Output]** Output the final CNF $\mathcal{C}$.

Let us show the work of Algorithm 1 on the set $\mathcal{C}$ of clauses defined by the CNF ϕ from Example 1. Let us assume that Algorithm 1 processes the strong components of the clause graph $\mathbf{D}_{\mathcal{C}}$ in the order K_1, K_2, K_3 and K_4 given by Fig. 3. Let us consider the processing of these four strong components step by step.

– Strong component $K = K_1$ is of type $\mathcal{K}_1$. We get $Q(K) = C_1$ and hence no clause in K has any subgoals outside of $Q(K)$. Thus no switching takes place in step (a) and we get $K = K'$ and $\mathcal{C} = \mathcal{C}'$. In step (b) the subgraph $Q'(K')$ is in fact the entire component $Q(K') = Q(K)$ as no clauses outside of $K' = K$ contribute arcs in $Q(K')$. Moreover, all arcs in $Q'(K')$ are marked as free since K' is a source component of $\mathbf{D}_{\mathcal{C}'}$ and hence $(Cone_{\mathbf{D}_{\mathcal{C}'}}(K') \setminus K')$ is empty. Finally, in steps (c) and (d) the four quadratic clauses in K' are replaced by three clauses (e.g. $\overline{x}_1 \vee x_2, \overline{x}_2 \vee x_3, \overline{x}_3 \vee x_1$) which span a directed cycle in $Q(K')$.

- When processing $K = K_2$ and $K = K_3$ which are also both of type $\mathcal{K}_1$ we get in both cases $Q(K) = C_2$ and no subgoals outside of $Q(K)$. Hence no switching occurs in step (a). For $K = K_2$ the subgraph $Q'(K')$ consists of a single arc (x_6, x_7) which is marked as free in step (b) and no change occurs in steps (c) and (d). For $K = K_3$ the subgraph $Q'(K')$ consists of arc (x_6, x_7) which is marked as fixed and arcs (x_5, x_4), (x_5, x_6) which is marked as free in step (b). Again, no change occurs in steps (c) and (d).
- The most interesting case occurs when processing $K = K_4$ which is also of type $\mathcal{K}_1$ and $Q(K) = C_2$. Each of the three cubic clauses in K has a different subgoal outside of $Q(K)$, namely x_1, x_2, and x_3. Therefore in step (a) these subgoals are all switched to the same subgoal, e.g. to x_1. See Fig. 4 for a directed hypergraph which corresponds to the CNF after the switch.
  In step (b) we get $Q'(K') = Q(K) = C_2$. Note that K is a terminal component of $\mathbf{D}_{\mathcal{C}'}$ and so all clauses in $\mathcal{C}'$ are from $Cone_{\mathbf{D}_{\mathcal{C}'}}(K')$. The arcs (x_6, x_7), (x_5, x_4), (x_5, x_6) are marked as fixed, while (x_4, x_5), (x_6, x_5), (x_7, x_6) are marked as free. Finally, in steps (c) and (d) the arcs (x_7, x_6), (x_6, x_5) are replaced by (x_7, x_5) which keeps all fixed arcs in place and maintains the strong component C_2. This implies replacing the cubic clauses $\overline{x}_1 \vee \overline{x}_7 \vee x_6$ and $\overline{x}_1 \vee \overline{x}_6 \vee x_5$ by a new cubic clause $\overline{x}_1 \vee \overline{x}_7 \vee x_5$. The resulting CNF

$$(\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge (\overline{x}_3 \vee x_1) \wedge$$

$$\wedge (\overline{x}_5 \vee x_4) \wedge (\overline{x}_5 \vee x_6) \wedge (\overline{x}_6 \vee x_7) \wedge (\overline{x}_1 \vee \overline{x}_4 \vee x_5) \wedge (\overline{x}_1 \vee \overline{x}_7 \vee x_5)$$

  is shown as a directed hypergraph in Fig. 5.

The rest of this section is organized as follows. Subsection 6.1 proves the correctness of Algorithm 1, i.e., it shows that Algorithm 1 outputs a minimum CNF representation of the CQ-function given by the input CNF. Subsection 6.2 shows that Algorithm 1 not only minimizes the number of clauses but it also outputs a minimum CNF with respect to the number of literals. Finally, Subsection 6.3 deals with the time complexity of Algorithm 1 and proves that it runs in $O(n^2 + m\ell)$ time where n is the number of variables, m is the number of clauses, and ℓ is the number of literals in the input CNF.

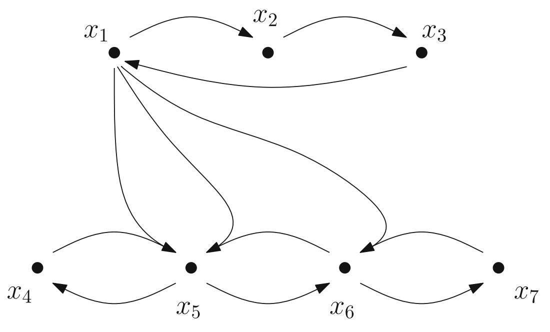

Fig. 4 Hypergraph corresponding to the CNF after the switch

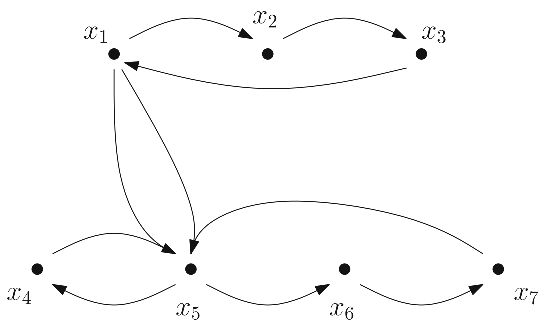

Fig. 5 Hypergraph corresponding to the resulting CNF produced by Algorithm 1 from CNF φ defined in Example 1

6.1 Correctness of algorithm 1

Theorem 11 *Algorithm* 1 *works correctly and outputs a minimum CNF representation of the given input function h.*

Proof Algorithm 1 follows the idea from the proof of Theorem 9. The only thing we have to prove is, that we are able to solve problem ORTSC if we restrict our attention to CQ functions. Action $\mathcal{K}_0$ solves problem ORTSC in case of strong components of type $\mathcal{K}_0$ and its correctness is justified by Lemma 21. Action $\mathcal{K}_1$ solves problem ORTSC in case of strong components of type $\mathcal{K}_1$ and its correctness is justified by Lemma 22.

In both lemmas we will use the following notation (it is same notation as in the proof of Theorem 9):

$$\begin{aligned} L_i &= \text{ strong component of graph } \mathbf{D}_h \text{ corresponding to } K_i \\ \mathcal{V} &= Cone_{\mathbf{D}_h}(K_i) \\ \mathcal{Y} &= Cone_{\mathbf{D}_h}(K_i) \cap \bigcup_{j=1}^{i-1} Cone_{\mathbf{D}_h}(K_j) \end{aligned}$$

Note that $Q = K_i$ and $\mathcal{F} = \mathcal{C} \cap \mathcal{Y} = Cone_{D_\mathcal{C}}(K_i) \setminus K_i$ in the notation from the definition of ORTSC. Moreover, note that $L_i \subseteq \mathcal{V} \setminus \mathcal{Y}$, where the inclusion is proper if $\mathcal{X}_i \setminus \mathcal{X}_{i-1}$ (where $\mathcal{X}_i = \bigcup_{j=1}^{i} Cone_{\mathbf{D}_h}(K_j)$) contains some redundant strong component of $\mathbf{D}_h$. □

Lemma 21 (Action $\mathcal{K}_0$) *Let K_i be of type $\mathcal{K}_0$, then K_i is a minimum cardinality set such that $\mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$.*

Proof We shall show that the irredundancy of $\mathcal{C}$ is in this case sufficient for the intersection $\mathcal{C} \cap L_i$ to have the minimum possible cardinality and no further action is required. Let us denote $K_i = \mathcal{C} \cap L_i = \{A_1 \vee u_1, \ldots, A_\ell \vee u_\ell\}$. By Corollary 9 we may switch the subgoal sets of all clauses in K_i to some representative set A, such that after this switch we get $K' = \mathcal{C}' \cap L_i = \{A \vee u_1, \ldots, A \vee u_\ell\}$ and $\mathcal{C}'$ is still a prime and irredundant representation of h. The irredundancy of $\mathcal{C}'$ now implies that for every $i \neq j$ we have $u_i \neq u_j$. Moreover, by Proposition 2 we have $\mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$.

Now let us assume by contradiction that there exists a set of clauses $P = \{B_1 \vee w_1, \ldots, B_\ell \vee w_{\ell'}\}$ where $\ell' < \ell$ such that $\mathcal{R}(\mathcal{Y} \cup P) = \mathcal{R}(\mathcal{V})$. By Proposition 2 and exclusiveness of $\mathcal{Y}$ and $\mathcal{V}$ it follows that replacing K' by P produces again a CNF representation of h and we may w.l.o.g. assume that P is such that this CNF is irredundant and prime. Now we may once more use Corollary 9 to switch the subgoal sets of all clauses in P to some representative set B, such that after this switch we get $P' = \mathcal{C}'' \cap L_i = \{B \vee w_1, \ldots, B \vee w_{\ell'}\}$ and $\mathcal{C}''$ is still a prime and irredundant representation of h. Clearly, after such a switch we again have $\mathcal{R}(\mathcal{Y} \cup P') = \mathcal{R}(\mathcal{Y} \cup P) = \mathcal{R}(\mathcal{V})$.

Due to irredundancy of $\mathcal{C}'$ and $\mathcal{C}''$ no clause in $K' \cup P'$ can be in $\mathcal{R}(\mathcal{Y})$. On the other hand $\mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{Y} \cup P')$ implies $K' \subseteq \mathcal{R}(\mathcal{Y} \cup P')$ and $P' \subseteq \mathcal{R}(\mathcal{Y} \cup K')$. Therefore any forward chaining derivation of a variable u_k, $k = 1, \ldots, \ell$, from the set A using the set of clauses $\mathcal{Y} \cup P'$ must involve at least one clause $B \vee w_j \in P'$ (and symmetrically any forward chaining derivation of a variable w_k from the set B using the set of clauses $\mathcal{Y} \cup K'$ must involve at least one clause $A \vee u_j \in K'$). Then, by the forward chaining Lemma 14, $w_j \in Cone_{G_{\mathcal{Y} \cup P'}}(u_k)$, and so there must be a path in $G_{\mathcal{Y} \cup P'}$ leading from w_j to u_k. Since L_i is of type $\mathcal{K}_0$, no arc in $Q(L_i)$ can be generated by clauses in $P' \subseteq L_i$. Therefore, for every u_k there exists a w_j such that there is a path in $G_{\mathcal{Y}}$ leading from w_j to u_k. Since $\ell' < \ell$, there is some w_j (w.l.o.g. w_1), from which paths lead to (at least) two different u_k's (w.l.o.g., to u_1 and u_2). However, by a symmetric argument, for every w_k (and in particular for w_1) there exists a u_j such that there is a path in $G_{\mathcal{Y}}$ leading from u_j to w_k. Thus there is a path in $G_{\mathcal{Y}}$ from some u_j to both u_1 and u_2, and so no matter what the value of j is, there is a path in $G_{\mathcal{Y}}$ from u_j to u_k where $j \neq k$. We shall finish the proof by showing that the existence of such a path contradicts the irredundancy of $\mathcal{C}'$.

Due to Theorem 3 we can assume that all arcs on the path from u_j to u_k are generated by clauses present in $\mathcal{C}' \cap \mathcal{Y}$, namely by some $B_1 \vee \overline{u}_i \vee w_1, B_2 \vee \overline{w}_1 \vee w_2, \ldots, B_{\ell-1} \vee \overline{w}_{\ell-2} \vee w_{\ell-1}, B_\ell \vee \overline{w}_{\ell-1} \vee u_j \in \mathcal{Y}$, where $B_i \cap Q(K') = \emptyset$ for every $i = 1, \ldots, \ell$. Since $A \vee u_j \in L_i$, $\mathcal{Y} \subseteq Cone_{\mathbf{D}_h}(L_i)$, and $\mathbf{D}_h$ is transitively closed, the graph $\mathbf{D}_h$ contains arcs $(B_i \vee \overline{w}_{i-1} \vee w_i, A \vee u_j)$ for every $i = 1, \ldots, \ell$ (here we denote $u_j = w_0$ and $u_k = w_\ell$), and therefore $B_i \subseteq FC_h(A)$ holds for each such index i. Let $I = Cone_{G_h}(Q(K')) \setminus Q(K')$. Since $A, B_i \subseteq I$ for all i, it follows from Lemma 15 that the inclusion $B_i \subseteq FC_{h|_I}(A)$ holds for each index i. Let us now define $\mathcal{C}^* = \mathcal{C}' \setminus \{A \vee u_k\}$, and let us investigate the set $FC_{\mathcal{C}^*}(A)$. First of all, since $u_k \notin I$, the inclusion $B_i \subseteq FC_{h|_I}(A) \subseteq FC_{\mathcal{C}^*}(A)$ holds for each index i. Secondly, since $A \vee u_j \in \mathcal{C}^*$, we get $B_1 \cup \{u_j\} \subseteq FC_{\mathcal{C}^*}(A)$ and so $w_1 \in FC_{\mathcal{C}^*}(A)$. Similarly, $B_2 \cup \{w_1\} \subseteq FC_{\mathcal{C}^*}(A)$ implies $w_2 \in FC_{\mathcal{C}^*}(A)$, and so on, eventually obtaining $u_k \in FC_{\mathcal{C}^*}(A)$. Thus $\phi(\mathcal{C}^*) \equiv \phi(\mathcal{C}')$ which is a contradiction to the irredundancy of $\mathcal{C}'$. □

Lemma 22 (Action $\mathcal{K}_1$) *Let K_i be of type $\mathcal{K}_1$, then the set K_i after being modified by Action $\mathcal{K}_1$ is a minimum cardinality set such that $\mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$.*

Proof Before we start the proof, let us describe its main steps. We shall start by proving that the set K' created in step (a) of Action $\mathcal{K}_1$ satisfies that $\mathcal{R}(\mathcal{Y} \cup$

$K') = \mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$ and that $Q(K_i) = Q(K')$. Then we shall show that the set K'', which denotes the set K_i after being modified by Action $\mathcal{K}_1$, satisfies that $\mathcal{R}(\mathcal{Y} \cup K'') = \mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{V})$. After that we shall show that $\mathcal{C}''$ which denotes the set $\mathcal{C} \cap \mathcal{V}$ after being modified by Action $\mathcal{K}_1$ is again the prime and irredundant representation of $\mathcal{V}$-component $h_{\mathcal{V}}$ of h. We will finish the proof by showing that K'' is a minimum cardinality set for which $\mathcal{R}(\mathcal{Y} \cup K'') = \mathcal{R}(\mathcal{V})$.

Let us denote $K_i = \mathcal{C} \cap L_i = \{A_1 \vee \overline{u}_1 \vee v_1, \dots, A_k \vee \overline{u}_k \vee v_k\}$ where $A_j \cap Q(K_i) = \emptyset$ and $\{u_j, v_j\} \subseteq Q(K_i)$ for each index j. By Corollary 9 we can switch all sets of subgoals outside of $Q(K_i)$ of all clauses from K_i to some representative set A, such that after this switch we obtain $K' = \mathcal{C}' \cap L_i = \{A \vee \overline{u}_1 \vee v_1, \dots, A \vee \overline{u}_k \vee v_k\}$. Corollary 9 guarantees that after this switch is performed in step (a) of Action $\mathcal{K}_1$ the "current" CNF $\mathcal{C}'$ is still a prime and irredundant representation of h. Note also that K' is now a strong component of $\mathbf{D}_{\mathcal{C}'}$ (such that L_i is its corresponding strong component of $\mathbf{D}_h$) and $\mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$. Moreover, since the switch did not change anything inside of $Q(K_i) = Q(K')$ the subgraphs of $Q(K')$ generated by clauses in K_i and K' are identical.

Let us investigate the exclusive set $\mathcal{V} = Cone_{\mathbf{D}_h}(K_i) = Cone_{\mathbf{D}_h}(K')$ of h and the corresponding exclusive component $h_{\mathcal{V}}$ of h. Since $\mathcal{C}'$ is a prime and irredundant representation of h, $\mathcal{C}' \cap \mathcal{V}$ is, due to Lemma 9, a prime and irredundant representation of $h_{\mathcal{V}}$. Moreover, the fact $\mathcal{I}(h) = \mathcal{R}(\mathcal{C}')$ implies by Lemma 16 that $\mathbf{D}_{\mathcal{C}'} = \mathbf{D}_{\mathcal{I}(h)}(\mathcal{C}') = \mathbf{D}_h(\mathcal{C}')$ which in turn implies $Cone_{\mathbf{D}_{\mathcal{C}'}}(K') = \mathcal{C}' \cap Cone_{\mathbf{D}_h}(K') = \mathcal{C}' \cap \mathcal{V}$.

In step (b) of Action $\mathcal{K}_1$ the graph $Q'(K')$ is defined as the subgraph of $Q(K')$ generated by clauses from $Cone_{\mathbf{D}_{\mathcal{C}'}}(K') = \mathcal{C}' \cap \mathcal{V}$. We shall show that if there is a path in $Q'(K')$ from u to v, then $A \vee \overline{u} \vee v \in \mathcal{R}(\mathcal{V})$ (i.e., $A \vee \overline{u} \vee v$ is an implicate of $h_{\mathcal{V}}$). Since $\mathcal{C}' \cap \mathcal{V}$ represents $h_{\mathcal{V}}$ this is the same as proving $A \vee \overline{u} \vee v \in \mathcal{R}(\mathcal{C}' \cap \mathcal{V}))$. The last fact is (by Lemma 2) equivalent to showing that $v \in FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$. Let $B_1 \vee \overline{u} \vee w_1, B_2 \vee \overline{w}_1 \vee w_2, \dots, B_k \vee \overline{w}_{k-1} \vee v$ be the clauses from $\mathcal{C}' \cap \mathcal{V} = Cone_{\mathbf{D}_{\mathcal{C}'}}(K')$ that generate a path from u to v in $Q'(K')$, where $w_1, \dots w_{k-1} \in Q(K')$. Since $A \vee \overline{u}_1 \vee v_1 \in K'$ (we could pick any clause from K' here) the arc $(B_\ell \vee \overline{w}_{\ell-1} \vee w_\ell, A \vee \overline{u}_1 \vee v_1)$ is in $\mathbf{D}_{\mathcal{C}'}$ for every $\ell = 1, \dots, k$ (here we denote $w_0 = u$ and $w_k = v$) by the definition of $Cone_{\mathbf{D}_{\mathcal{C}'}}(K')$ and the fact that $\mathbf{D}_{\mathcal{C}'}$ is transitively closed. However, the definition of the graph $\mathbf{D}_{\mathcal{C}'}$ now implies $B_\ell \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u_1\})$ for every $\ell = 1, \dots, k$. However, $B_\ell \subseteq Cone_{G_h}(Q(K')) \setminus Q(K')$ and so by Lemma 14 no clause with literals in $Q(K')$ (and in particular no clause containing u_1) is necessary to derive B_ℓ by forward chaining from $A \cup \{u_1\}$. Hence $B_\ell \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A)$ for every $\ell = 1, \dots, k$. This means that $B_1 \cup \{u\} \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$ which together with $B_1 \vee \overline{u} \vee w_1 \in \mathcal{C}' \cap \mathcal{V}$ gives $w_1 \in FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$. Similarly, $B_2 \cup \{w_1\} \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$ together with $B_2 \vee \overline{w}_1 \vee w_2 \in \mathcal{C}' \cap \mathcal{V}$ gives $w_2 \in FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$ and so on, eventually obtaining $v \in FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$, which finishes the proof that $A \vee \overline{u} \vee v \in \mathcal{R}(\mathcal{V})$.

Note that in step (c) an arc (u, v) is added into the transitive reduction of $Q'(K')$ only if there is a path in $Q'(K')$ from u to v. Therefore the argument in the previous paragraph proves that every clause added in step (d) is in $\mathcal{R}(\mathcal{V})$. Thus $\mathcal{R}(\mathcal{Y} \cup K'') \subseteq \mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{V})$ holds after K' is modified to K'' in step (d). Let us denote $\mathcal{C}_{old} = (\mathcal{C}' \cap \mathcal{V}) \setminus (K' \setminus K'')$, $\mathcal{C}_{new} = (K'' \setminus K')$ and $\mathcal{C}'' = \mathcal{C}_{old} \cup \mathcal{C}_{new}$. To prove the reverse inclusion $\mathcal{R}(\mathcal{Y} \cup K') \subseteq \mathcal{R}(\mathcal{Y} \cup K'')$ we need to show that for every clause $A \vee \overline{u} \vee v$ removed from K' in step (d), v can be derived by forward chaining from $A \cup \{u\}$ using the new set of clauses $\mathcal{C}''$. Note that an arc (u, v) is removed from $Q'(K')$ in step (c) only if there is a path from u to v in the constructed transitive reduction which means that clauses in $\mathcal{C}''$ generate a path from u to v. However, now we can repeat the argument from the previous paragraph to prove that $v \in FC_{\mathcal{C}''}(A \cup \{u\})$ provided we can show $B_\ell \subseteq FC_{\mathcal{C}''}(A)$ for every clause $B_\ell \vee \overline{w}_{\ell-1} \vee w_\ell \in \mathcal{C}''$ generating an arc on the path from u to v. If $B_\ell \vee \overline{w}_{\ell-1} \vee w_\ell \in \mathcal{C}_{old}$ then we already know that $B_\ell \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A)$. However, $B_\ell \subseteq Cone_{G_h}(Q(K')) \setminus Q(K')$ and so by Lemma 14 no clause with its head in $Q(K')$ is necessary to derive B_ℓ by forward chaining from A. Hence $B_\ell \subseteq FC_{(\mathcal{C}' \cap \mathcal{V}) \setminus K'}(A)$ which implies $B_\ell \subseteq FC_{\mathcal{C}''}(A)$ as $\mathcal{C}' \setminus K'$ and $\mathcal{C}'' \setminus K''$ are identical. If $B_\ell \vee \overline{w}_{\ell-1} \vee w_\ell \in \mathcal{C}_{new}$ then $B_\ell \subseteq FC_{\mathcal{C}''}(A)$ holds trivially as $B_\ell = A$. This finishes the proof of $\mathcal{R}(\mathcal{Y} \cup K'') = \mathcal{R}(\mathcal{Y} \cup K')$.

The use of Corollary 9 in step (a) at every iteration of the algorithm requires not only that all added clauses be implicates of $h_\mathcal{V}$ (and thus implicates of h) but also that all added clauses be prime implicates, maintaining the primality of the CNF representation of h. Let us assume that an arc (u, v) is added into the transitive reduction of $Q'(K')$ in step (c). Since the minimum possible number of arcs has been added to get back the same transitive closure, it implies that all paths from u to v in $Q'(K')$ contain at least one free arc which is then removed in step (d) (note that no fixed arcs are removed and no arc spanning a path which entirely stays in $Q'(K')$ can be added into its transitive reduction). The addition of the arc (u, v) in step (c) implies the addition of the clause $A \vee \overline{u} \vee v$ in step (d). Let us show that $A \vee \overline{u} \vee v$ is a prime implicate of $h_\mathcal{V}$. Let us distinguish two cases:

- First let us assume by contradiction that $A \vee v$ is an implicate of $h_\mathcal{V}$, and hence $\mathcal{C}^* = \mathcal{C}'' \setminus \{A \vee \overline{u} \vee v\} \cup \{A \vee v\}$ still represents $h_\mathcal{V}$. However, the arc (u, v) which was added in step (c) is now missing from $Q'(K')$, and since the minimum number of arcs was added to keep the transitive closure of $Q'(K)$ the same, the transitive closure will change as a result of removing (u, v). That means that the graphs $\mathbf{G}_{\mathcal{C}''}$ and $\mathbf{G}_{\mathcal{C}^*}$ have different transitive closures, which by Theorem 3 contradicts the assumption that $\mathcal{C}''$ represents the same function as $\mathcal{C}^*$. This argument also proves that $\mathcal{C}''$ is an irredundant representation of $h_\mathcal{V}$ and thus $\mathcal{C}' \setminus (\mathcal{C}' \cap \mathcal{V}) \cup \mathcal{C}''$ is an irredundant representation of h.
- Now let us assume by contradiction that $A' \vee \overline{u} \vee v$, where $A' \subset A$, is an implicate of $h_\mathcal{V}$, and hence $\mathcal{C}^* = \mathcal{C}'' \setminus \{A \vee \overline{u} \vee v\} \cup \{A' \vee \overline{u} \vee v\}$ still represents $h_\mathcal{V}$. This implies that $v \in FC_{\mathcal{C}' \cap \mathcal{V}}(A' \cup \{u\})$ while $v \notin FC_{\mathcal{C}' \cap \mathcal{V}}(A')$ due to the above first case. Therefore, any forward chaining derivation of v from $A' \cup \{u\}$ must use some clause C_1 in which u is a subgoal. By Lemma 14 also the head (say w_1) of this clause must be in $Q(K')$, and so we can write $C_1 = B_1 \vee \overline{u} \vee w_1$ for some $B_1 \cap Q(K') = \emptyset$ (since C_1 is a CQ clause). Assuming that we are looking at a forward chaining derivation of v from $A' \cup \{u\}$ in which all derived variables are subsequently used in some future step, also w_1 must be a subgoal of some clause C_2 used in the derivation chain. By similar considerations as above we can write $C_2 = B_2 \vee \overline{w}_1 \vee w_2$ where $w_2 \in Q(K')$ and $B_2 \cap Q(K') = \emptyset$, and so on, until some clause $C_\ell = B_\ell \vee \overline{w}_{\ell-1} \vee v$ is used. Clauses C_1 to C_ℓ of course generate a path from u to v in $Q'(K')$. However, every such path contains at least one free arc generated by a clause from K' and so $A \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A' \cup \{u\})$ must hold. But now using the fact that $A \subseteq Cone_{G_h}(Q(K')) \setminus Q(K')$ and Lemma 14 we get that no clause with literals in $Q(K')$ (and in particular no clause containing u) is necessary to derive A by forward chaining from A' and so $A \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A')$ contradicting the primality of C'.

The above considerations show that the CNF $\mathcal{C}'' = ((\mathcal{C}' \cap \mathcal{V}) \setminus K') \cup K''$ resulting from Action $\mathcal{K}_1$ is an irredundant and prime representation of $h_{\mathcal{V}}$ such that $\mathcal{R}(\mathcal{Y} \cup K'') = \mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{V})$. Using Corollary 4 we moreover get that $(\mathcal{C}' \setminus (\mathcal{C}' \cap \mathcal{V})) \cup \mathcal{C}''$ is an irredundant and prime representation of h. To finish the proof we shall show that K'' is a minimum cardinality set with this property.

Let us assume by contradiction that there exists a set of clauses K^*, where $|K^*| < |K''|$, such that $\mathcal{R}(\mathcal{Y} \cup K^*) = \mathcal{R}(\mathcal{Y} \cup K'')$. Let $Q''(K')$ and $Q^*(K')$ be the subgraphs of $Q(K')$ generated by clauses from $\mathcal{Y} \cup K''$ and $\mathcal{Y} \cup K^*$. Since the set of arcs generated by clauses from $\mathcal{Y}$ is exactly the same in both cases, and the set of arcs K'' has the minimum cardinality subject to the condition that the transitive closure of $Q''(K')$ is the same as that of $Q'(K')$, then the transitive closure of $Q^*(K')$ must be different. That means that the graphs $\mathbf{G}_{\mathcal{Y} \cup K''}$ and $\mathbf{G}_{\mathcal{Y} \cup K^*}$ have different transitive closures, which by Theorem 3 contradicts the assumption that $\mathcal{Y} \cup K''$ represents the same function as $\mathcal{Y} \cup K^*$. □

6.2 Minimization of the number of literals

In this subsection we shall show that given a CQ function f Algorithm 1 not only outputs a minimum CNF with respect to the number of clauses, but if a simple preprocessing and postprocessing step is employed (adding and removing an extra variable which turns an arbitrary CQ CNF into a CQ CNF with no negative clauses) then the output CNF is also minimum with respect to the number of literals. Throughout this subsection let us simply denote the clause graph D_f by D and let K be an arbitrary strong component of D. We shall show that all prime implicates of f which belong to K have the same degree, i.e., consist of the same number of literals. We shall start with two preparatory lemmas.

Lemma 23 *Let f be a CQ function and let $C = A \vee X \vee u$ be a prime implicate of f, let Q be the strong component of $G_{Cone_D(C)}$ to which u belongs and let us assume that $A \cap Q = \emptyset$, $X \subseteq Q \setminus \{u\}$, $|X| \leq 1$. Let K denote the strong component of $D = D_f$, to which C belongs. Then each strong component of $G_{Cone_D(C) \setminus K}$ contains at most one element of A.*

Proof Let us denote $\mathcal{X} = Cone_D(C) \setminus K$. According to Theorem 6, we have that $\mathcal{X}$ is an exclusive set of clauses of f. To prove the lemma, let us proceed by contradiction. Let us assume, that there are two distinct variables $a_1, a_2 \in A$ which belong to the same strong component S of $G_{\mathcal{X}}$. Since there is a path from a_1 to a_2 in $G_{\mathcal{X}}$ there must exist clauses $C_1 = B_1 \vee \overline{x}_1 \vee x_2, C_2 = B_2 \vee \overline{x}_2 \vee x_3, \ldots, C_k = B_k \vee \overline{x}_k \vee x_{k+1}$ in $\mathcal{X}$, where $x_1 = a_1$, $x_{k+1} = a_2$, and $\forall\, 1 \leq i \leq k+1 : x_i \in S$. The following properties can now be derived for every $1 \leq i \leq k$:

- From the fact that $\mathcal{X} \subseteq \mathcal{I}(f)$ and from Corollary 1 it follows that C_i is a CQ-clause w.r.t. f, which in turn implies that $B_i \subseteq Cone_{G_{\mathcal{X}}}(S) \setminus S$.
- Moreover, since $C_i \in \mathcal{X} = Cone_D(C) \setminus K$, we get $B_i \subseteq FC_{\mathcal{X}}(A \cup X)$ by the definition of graph D, and using Lemma 15 (where we set $I = Cone_{G_{\mathcal{X}}}(S) \setminus S$) we can strengthen this to $B_i \subseteq FC_{\mathcal{X}}(A)$.

Let us denote $B = \bigcup_{i=1}^{k} B_i$. Clearly, the properties of individual B_i's carry over to set B and so $B \subseteq Cone_{G_{\mathcal{X}}}(S) \setminus S$ and $B \subseteq FC_{\mathcal{X}}(A)$ hold. Using the resolution chain of

C_i clauses along the path from a_1 to a_2 also $C' = B \vee \overline{a}_1 \vee a_2 \in \mathcal{R}(\mathcal{X}) \subseteq \mathcal{I}(f)$ holds. Now there are two possibilities:

- Either $B \subseteq FC_{\mathcal{X}}(A \setminus \{a_2\})$ but then using the clause $C' \in \mathcal{I}(f)$ we get $a_2 \in FC_f(A \setminus \{a_2\})$, contradicting the primality of C,
- or there must exist a variable $b \in B$ such that each forward chaining derivation proving $b \in FC_{\mathcal{X}}(A)$ uses a_2. Let $\{E_1, \ldots, E_\ell\} \subseteq \mathcal{X}$ be a minimal (under inclusion) forward chaining derivation of $b \in FC_{\mathcal{X}}(A)$, then due to the above considerations $a_2 \in \bigcup_{j=1}^{k} Vars(E_j)$ must hold. However, Lemma 14 now implies that there is a path from a_2 to b in $G_{\mathcal{X}}$ contradicting the fact that $B \subseteq Cone_{G_{\mathcal{X}}}(S) \setminus S$. □

Lemma 24 *Let f be a CQ function and let $C = A \vee X \vee u$ and $C' = B \vee Y \vee v$ be two prime implicates of f that belong to the same strong component K of clause graph $D = D_f$. Let Q be the strong component of G_f to which both u and v belong, and let us assume that $A \cap Q = \emptyset$, $X \subseteq Q \setminus \{u\}$, $|X| \leq 1$, $B \cap Q = \emptyset$, $Y \subseteq Q \setminus \{v\}$, $|Y| \leq 1$. Let S be an arbitrary strong component of $G_{Cone_D(C) \setminus K}$. Then $|S \cap A| = |S \cap B| \leq 1$.*

Proof Similarly as in the proof of Lemma 23 let us denote the exclusive set of clauses $Cone_D(C) \setminus K = Cone_D(C') \setminus K$ by $\mathcal{X}$. Using Lemma 20 we get $A \subseteq FC_{\mathcal{X}}(B)$ and $B \subseteq FC_{\mathcal{X}}(A)$. By Lemma 23, $|S \cap A| \leq 1$ and $|S \cap B| \leq 1$, so we only need to show that if $|S \cap A| = 1$ (say $S \cap A = \{a\}$) then there exists some $b \in S \cap B$ (the opposite direction is of course similar).

Now $A \subseteq FC_{\mathcal{X}}(B)$ implies $a \in FC_{\mathcal{X}}(B)$, i.e., $B \vee a$ is an implicate of function $f_{\mathcal{X}}$, the $\mathcal{X}$-component of f. Therefore there must exist $B' \subseteq B$ such that $B' \vee a$ is a prime implicate of $f_{\mathcal{X}}$, which implies $a \in FC_{\mathcal{X}}(B')$. Using Theorem 3 there must be a path from every variable $b \in B'$ to a in the graph $G_{\mathcal{X}}$. Furthermore, the inclusion $B \subseteq FC_{\mathcal{X}}(A)$ of course implies $B' \subseteq FC_{\mathcal{X}}(A)$, and so, similarly as in the proof of Lemma 23, there are two possibilities:

- Either $B' \subseteq FC_{\mathcal{X}}(A \setminus \{a\})$ but then also $a \in FC_f(A \setminus \{a\})$ (recall that $a \in FC_{\mathcal{X}}(B')$), contradicting the primality of C,
- or there must exist a variable $b \in B'$ such that each forward chaining derivation proving $b \in FC_{\mathcal{X}}(A)$ uses a. Let $\{E_1, \ldots, E_\ell\} \subseteq \mathcal{X}$ be a minimal (under inclusion) forward chaining derivation of $b \in FC_{\mathcal{X}}(A)$, then due to the above considerations $a \in \bigcup_{j=1}^{k} Vars(E_j)$ must hold. However, Lemma 14 now implies that there is a path from a to b in $G_{\mathcal{X}}$. Thus $G_{\mathcal{X}}$ contains both a path from b to a and a path from a to b proving that $b \in S$. □

Now we are ready to derive the desired result.

Corollary 12 *Let f be a CQ function and let $C = A \vee u$ and $C' = B \vee v$ be two prime implicates of f, which belong to the same strong component K of $D = D_f$, then $|A| = |B|$.*

Proof Let Q be the strong component of G_f which contains both u and v. According to Corollary 11 both C and C' are of the same type (either $\mathcal{K}_0$ or $\mathcal{K}_1$) and hence $|A \cap Q| = |B \cap Q|$. According to Lemma 24, $|S \cap A| = |S \cap B|$ for every strong component S of the graph $G_{Cone_D(C)\setminus K}$, and hence $|A \setminus Q| = |B \setminus Q|$. Therefore $|A| = |B|$. □

As an easy corollary, we now get that Algorithm 1 for minimizing the number of clauses in a CQ CNF minimizes also the number of literals, if the input CNF is pure Horn.

Corollary 13 *Let f be a pure Horn CQ function and let $\mathcal{C}$ be a CNF with the minimum number of clauses which represents f and which was produced by Algorithm* 1. *Then $\mathcal{C}$ is also a CNF with the minimum number of literals which represents f.*

Proof Algorithm 1 in each step solves one ORTSC problem by minimizing the number of clauses in some strong component K of D_f. Using Corollary 12 we get that all prime clauses in K have the same number of literals. It then follows that a minimum cardinality set $Q^* \subseteq K$ which solves the current ORTSC is also a set with the minimum number of literals which solves the ORTSC reformulated to output a set $\mathcal{Q}^*$ consisting of a minimum number of literals such that $\mathcal{R}(\mathcal{F} \cup \mathcal{Q}^*) = \mathcal{R}(\mathcal{F} \cup \mathcal{Q})$ (as in Remark 1). □

Using a simple transformation we can extend Corollary 13 to all CQ functions including those that have negative prime implicates.

Corollary 14 *Let f be a CQ function given by a prime and irredundant CQ CNF $\mathcal{F}$. Then a CNF with the minimum number of literals which represents f can be constructed using a simple modification of Algorithm* 1.

Proof If $\mathcal{F}$ is pure Horn then Algorithm 1 constructs the desired CNF due to Corollary 13. If $\mathcal{F}$ contains negative clauses then introduce a new variable z and replace each negative clause $\bigvee_{i\in I} \overline{x}_i$ in $\mathcal{F}$ with a pure Horn clause $\bigvee_{i\in I} \overline{x}_i \vee z$. Notice that the new pure Horn CNF $\mathcal{F}_z$ resulting from such a replacement is CQ, because all newly introduced arcs in $G_{\mathcal{F}_z}$ point to z, which means that z constitutes a new singleton strong component of $G_{\mathcal{F}_z}$, while the structure of all the other strong components of $G_{\mathcal{F}_z}$ remains the same as in $G_{\mathcal{F}}$. Let f_z be the CQ function represented by $\mathcal{F}_z$. Note that $f_z(z = 0) = f$, and $f_z(z = 1) = f_{\mathcal{H}}$, where $f_{\mathcal{H}}$ is the pure Horn component of f.

Let $\mathcal{F}'_z$ be a CNF produced by Algorithm 1 on input $\mathcal{F}_z$. Using Corollary 13 we get that $\mathcal{F}'_z$ is a representation of f_z with the minimum number of literals. Let $\mathcal{F}'$ be the CNF produced from $\mathcal{F}'_z$ by dropping all occurrences of literal z (or equivalently, by substituting $z = 0$). We claim that $\mathcal{F}'$ is a representation of f with the minimum number of literals.

Note first that $\mathcal{F}'$ represents f, since $\mathcal{F}' = \mathcal{F}'_z(z = 0)$, and $\mathcal{F}'_z$ represents f_z and $f_z(z = 0) = f$. Let k be the number of negative clauses in $\mathcal{F}$. Let us assume by contradiction that $\mathcal{F}''$ is a prime and irredundant representation of f with fewer literals than $\mathcal{F}'$. By Proposition 4 both $\mathcal{F}'$ and $\mathcal{F}''$ have exactly k negative clauses. Let $\mathcal{F}''_z$ be the CNF constructed from $\mathcal{F}''$ by adding z to all negative clauses, as above. Clearly, $\mathcal{F}''_z$ represents f_z since $\mathcal{F}''_z(z = 0)$ represents f and $\mathcal{F}''_z(z = 1)$ represents $f_{\mathcal{H}}$. However, now we have: $|\mathcal{F}''_z|_\ell = |\mathcal{F}''|_\ell + k < |\mathcal{F}'|_\ell + k = |\mathcal{F}'_z|_\ell$ contradicting the minimality of $\mathcal{F}'_z$. □

6.3 Time complexity of algorithm 1

Now let us turn our attention to the complexity of CQ minimization. First let us recall that due to Theorem 2, we can transform any given Horn CNF representation (having ℓ literals) of a CQ function h into an irredundant and prime CNF $\mathcal{C}$ in $O(\ell^2)$ time. By Lemma 4 we know that such a prime representation must be a CQ CNF, which can be easily tested in $O(\ell)$ time (by building the strong components of $G_{\mathcal{C}}$ in $O(\ell)$ time, and then testing whether each clause is CQ in $O(\ell)$ time). Once we have an irredundant and prime CNF representation of a CQ function, we can run Algorithm 1.

Theorem 12 *Let h be a Horn function on n variables which is from the class CQ. Let $\mathcal{C}$ be an irredundant and prime CNF representing h which is given as an input to Algorithm* 1. *Then Algorithm* 1 *runs in $O(n^2 + m\ell)$ time, where m is the number of clauses and ℓ is the number of literals in $\mathcal{C}$.*

Proof As we have shown in property (iv) of Theorem 8, we can perform the first and the second step in time $O(n^2 + m\ell)$. To classify the components according to Definition 11 (types $\mathcal{K}_0$ and $\mathcal{K}_1$) it suffices to look at one clause in $\mathcal{C}$ from each strong component, which can also be easily done in $O(m\ell)$ time. Note that $\mathbf{D}_{\mathcal{C}}$ is transitively closed, so it is sufficient to keep the graph of the acyclic condensation of $\mathbf{D}_{\mathcal{C}}$ and for each strong component a list of participating clauses.

The third step is the main loop and it is executed for each strong component of $D_{\mathcal{C}}$ exactly once. Thus the body of the loop is repeated at most m times. Performing Action $\mathcal{K}_0$ of course takes constant time. We shall show that Action $\mathcal{K}_1$ can be performed in $O(\ell)$ time for each strong component of type $\mathcal{K}_1$. Let us analyze Action $\mathcal{K}_1$ in detail.

- Step (a): switching all subgoal sets outside of $Q(K)$ of all clauses in $\mathcal{C} \cap K$ to some representative set $A(K)$ can be easily performed in $O(\ell)$ time.
- Step (b): identifying the subgraph $Q'(K')$ of $Q(K')$ generated only by clauses from $Cone_{D_{\mathcal{C}'}}(K')$ also takes only $O(\ell)$ time.
- Step (c): finding a transitive reduction of $Q'(K')$ with the added restriction that the reduction must contain all fixed arcs can be done in three steps:

  1. Detecting the strong components of $Q'(K')$ takes $O(|Q'(K')|)$ time (which is of course $O(\ell)$).
  2. Arcs between different strong components all stay in place. This is due to the fact that the irredundancy of the CNF representation $\mathcal{C}'$ guarantees that these arcs already span the skeleton graph of the acyclic condensation of $Q'(K')$ which is of course the unique transitive reduction of the acyclic condensation of $Q'(K')$. Therefore this step takes no time. To see that the irredundancy of $\mathcal{C}'$ is sufficient, let us assume by contradiction that there is a free arc (u, v) (generated by a clause C' in K') between two distinct strong components of $Q'(K')$ which is not in the skeleton graph of the acyclic condensation. That means that there is a path from u to v in $Q'(K')$ (i.e., a path generated by clauses in $Cone_{D_{\mathcal{C}'}}(K')$) not using the arc (u, v). By the definition of the graph $D_{\mathcal{C}'}$, for each such clause C'' we have $Subg(C'') \subseteq FC_{\mathcal{C}'}(Subg(C'))$. In fact, by Lemma 14 all such

forward chaining derivations can be done without ever using a clause containing v (in particular, without using the clause C'), and so $Subg(C'') \subseteq FC_{\mathcal{C}' \setminus C'}(Subg(C'))$. However, this means that $v \in FC_{\mathcal{C}' \setminus C'}(Subg(C'))$ (i.e., v can be derived from $Subg(C')$ along the path from u to v without using the clause C') which is a contradiction to the irredundancy of $\mathcal{C}'$.

3. In each strong component of $Q'(K')$ its transitive reduction that keeps all the fixed arcs can be found in $O(|Q'(K')|)$ (and hence also $O(\ell)$) time using the algorithm from [12] and its correction from [28].

– Step (d): removing clauses that correspond to the removed arcs and adding clauses that correspond to the added arcs in step (c) amounts to updating the list that represents the strong component K', which can be performed in $O(\ell)$ time as the total length of the added clauses is at most equal to the total length of the removed clauses (this also takes care of updating the graph $\mathbf{D}_{\mathcal{C}'}$).

The fourth and last step takes only $O(\ell)$ time, so adding all time requirements together we get the desired $O(n^2 + m\ell)$ time bound. Moreover, if we assume that each variable appears as a head of some clause, then the complexity of Algorithm 1 can be simplified to $O(m\ell)$ time. □

6.4 Compression performance of algorithm 1

By carefully examining Algorithm 1 it is not too hard to give an estimate on the maximum compression rate (in terms of the number of clauses) that the algorithm may achieve (recall that a prime and irredundant input CNF is assumed). Clauses are deleted and replaced by other clauses only in the Action $\mathcal{K}_1$ step. The subgoal set outside of the strong component of the implication graph plays no role in the minimization. The only procedure that is essential for the minimization of the input CNF is removing arcs from the strong component of the implication graph and replacing them with a smaller number of arcs while maintaining the strong connectivity of the component. Each arc in this procedure corresponds to a clause. It is not hard to see that the maximum number of arcs in a strong component on k vertices when no transitive (redundant) arcs are allowed is $2(k-1)$. Any spanning tree on k vertices where each edge of the tree is replaced by arcs in both directions provides such a directed graph. On the other hand, the minimum number of arcs that still maintain strong connectivity of the component is k. Any directed simple cycle through all k vertices provides such a graph. Therefore, the ratio of the maximum and minimum number of arcs in each strong component is bounded from above by two.

7 Conclusions

This paper introduces a new subclass of Horn functions - the class of CQ (component-wise quadratic) functions, which strictly contains the class of quasi-acyclic functions defined in [19]. For this new class a polynomial time algorithm is designed, which for a given prime and irredundant input CNF of a CQ function f outputs a minimum (shortest) CNF representation of f. The algorithm minimizes the number of clauses and, with an additional preprocessing step, can be also used to minimize the number of literals.

Moreover, the algorithm in fact works for a broader subclass of Horn functions than just the class of CQ functions. The property which suffices for the correctness of the algorithm is the following: every prime implicate $C = A \vee u$ of f has at most one subgoal from A in the strong component $Q_{Cone_{D_f}(C)}(u)$ (i.e., the strong component which contains u) of the implication graph induced by the exclusive component of f represented by clauses in $Cone_{D_f}(C)$. Since this implication graph is a subgraph of G_f (has fewer edges), the strong component $Q_{Cone_{D_f}(C)}(u)$ can be strictly smaller than the strong component $Q_f(u)$ in G_f. Thus, it may happen that some prime implicate $C = A \vee u$ has two or more subgoals in $Q_f(u)$ (which means that f is not a CQ function) while it still has at most one subgoal in $Q_{Cone_{D_f}(C)}(u)$ (thus, still enabling the minimization algorithm to work).

Aside of the theoretical contribution, it would also be interesting to assess the practical importance of the minimization algorithm. To that end, it would be interesting to find out how frequently the CQ functions may appear, or more specifically, to answer the following questions. Among the Horn functions over n variables, how many are component-wise quadratic? Among the component-wise quadratic Horn functions, how many are quasi-acyclic? These may be hard questions to solve analytically. Perhaps an asymptotic behavior of the corresponding ratios can be estimated through experiments (randomly generated Horn CNFs).

Acknowledgements We would like to thank two anonymous reviewers for their valuable comments which helped to improve this paper. The second author gratefully acknowledges the support by the Czech science Foundation (grant P202/10/1188). The fourth author gratefully acknowledges the support by the Czech Science Foundation (grant 201/07/P168).

References

1. Ausiello, G., D'Atri, A., Sacca, D.: Minimal representation of directed hypergraphs. SIAM J. Comput. **15**, 418–431 (1986)
2. Boros, E., Čepek, O.: On the complexity of Horn minimization. RUTCOR Research Report RRR 1-94, Rutgers University, New Brunswick, NJ (1994)
3. Boros, E., Čepek, O., Kogan, A.: Horn minimization by iterative decomposition. Ann. Math. Artif. Intell. **23**, 321–343 (1998)
4. Boros, E., Čepek, O., Kogan, A., Kučera, P.: Exclusive and essential sets of implicates of Boolean functions. Discrete Appl. Math. **158**(2), 81–96 (2010)
5. Buning, H.K., Letterman, T.: Propositional Logic: Deduction and Algorithms. Cambridge University Press (1999)
6. Cadoli, M., Donini, F.M.: A survey on knowledge compilation. AI Commun. **10**(3–4), 137–150 (1997)
7. Čepek, O.: Structural properties and minimization of Horn Boolean functions. Doctoral dissertation, Rutgers University, New Brunswick, NJ (1995)
8. Darwiche, A., Marquis, P.: A knowledge compilation map. J. Artif. Intell. Res. **17**, 229–264 (2002)
9. Dechter, R., Pearl, J.: Structure identification in relational data. Artif. Intell. **58**, 237–270 (1992)
10. Delobel, C., Casey, R.G.: Decomposition of a data base and the theory of Boolean switching functions. IBM J. Res. Develop. **17**, 374–386 (1973)
11. Dowling, W.F., Gallier, J.H.: Linear time algorithms for testing the satisfiability of propositional Horn formulae. J. Log. Program. **3**, 267–284 (1984)
12. Eswaran, K.P., Tarjan, R.E.: Augmentation problems. SIAM J. Comput. **5**, 653–665 (1976)
13. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, San Francisco (1979)

14. Genesereth, M.R., Nilsson, N.J.: Logical Foundations of Artificial Intelligence. Morgan Kaufmann, Los Altos, CA (1987)
15. Hammer, P.L., Kogan, A.: Horn functions and their DNFs. Inf. Process. Lett. **44**, 23–29 (1992)
16. Hammer, P.L., Kogan, A.: Horn function minimization and knowledge compression in production rule bases. RUTCOR Research Report RRR 8-92, Rutgers University, New Brunswick, NJ (1992)
17. Hammer, P.L., Kogan, A.: Optimal compression of propositional Horn knowledge bases: complexity and approximation. Artif. Intell. **64**, 131–145 (1993)
18. Hammer, P.L., Kogan, A.: Knowledge compression—logic minimization for expert systems. In: Computers As Our Better Partners. Proceedings of the IISF/ACM Japan International Symposium, pp. 306–312. World Scientific, Singapore (1994)
19. Hammer, P.L., Kogan, A.: Quasi-acyclic propositional Horn knowledge bases: optimal compression. IEEE Trans. Knowl. Data Eng. **7**(5), 751–762 (1995)
20. Ibaraki, T., Kogan, A., Makino, K.: Functional dependencies in Horn theories. Artif. Intell. **108**(1–2), 1–30 (1999)
21. Ibaraki, T., Kogan, A., Makino, K.: On functional dependencies in q-Horn theories. Artif. Intell. **131**(1–2), 171–187 (2001)
22. Itai, A., Makowsky, J.A.: Unification as a complexity measure for logic programming. J. Log. Program. **4**, 105–117 (1987)
23. Kautz, H., Kearns, M., Selman, B.: Forming concepts for fast inference. In: Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI'92), pp. 786–793. AAAI, San Jose, CA (1992)
24. Maier, D.: Minimal covers in the relational database model. J. ACM **27**, 664–674 (1980)
25. Minoux, M.: LTUR: a simplified linear time unit resolution algorithm for Horn formulae and computer implementation. Inf. Process. Lett. **29**, 1–12 (1988)
26. Quine, W.: The problem of simplifying the truth functions. Am. Math. Mon. **59**, 521–531 (1952)
27. Quine, W.: A way to simplify truth functions. Am. Math. Mon. **62**, 627–631 (1955)
28. Raghavan, S.: A note on Eswaran and Tarjan's algorithm for the strong connectivity augmentation problem. In: Golden, B.L., Raghavan, S., Wasil, E.A. (eds.) The Next Wave in Computing, Optimization, and Decision Technologies, pp. 19–26. Springer (2005)
29. Reiter, R., de Kleer, J.: Foundations of assumption-based truth maintenance systems: preliminary report. In: Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI'87), pp. 183–189. AAAI, San Jose, CA (1987)
30. Selman, B., Kautz, H.: Knowledge compilation and theory approximation. J. ACM **43**(2), 193–224 (1996)
31. Tarjan, R.E.: Depth first search and linear graph algorithms. SIAM J. Comput. **2**, 146–160 (1972)

Ann Math Artif Intell (2011) 61:231–244
DOI 10.1007/s10472-011-9263-9

Disjoint essential sets of implicates of a CQ Horn function

Ondřej Čepek · Petr Kučera

Published online: 12 November 2011

Abstract In this paper we study a class of CQ Horn functions introduced in Boros et al. (Ann Math Artif Intell 57(3–4):249–291, 2010). We prove that given a CQ Horn function f, the maximal number of pairwise disjoint essential sets of implicates of f equals the minimum number of clauses in a CNF representing f. In other words, we prove that the maximum number of pairwise disjoint essential sets of implicates of f constitutes a tight lower bound on the size (the number of clauses) of any CNF representation of f.

Keywords Horn functions · Boolean minimization · CQ functions · Essential sets · Exclusive sets

Mathematics Subject Classification (2010) 93C10

1 Introduction

The problem of CNF minimization can be stated as follows: given a CNF $\mathcal{F}$ find a logically equivalent CNF $\mathcal{F}'$ which consists of the minimum possible number of clauses. This problem has many practical applications in artificial intelligence [9, 13, 14] and database design [10, 16]. CNF minimization is a very hard problem: it is Σ_2^p-complete if the input can be any CNF [20, 21] and it is NP-complete (i.e.,

O. Čepek (✉) · P. Kučera
Department of Theoretical Computer Science and Mathematical Logic,
Charles University, Malostranské nám. 25, 118 00 Praha 1, Czech Republic
e-mail: ondrej.cepek@mff.cuni.cz

P. Kučera
e-mail: kucerap@ktiml.mff.cuni.cz

O. Čepek
Institute of Finance and Administration - VŠFS, Estonská 500,
100 00 Praha 10, Czech Republic

Σ_1^p-complete) if the input is restricted to Horn CNFs (see [1, 2, 7, 13, 16] for various Boolean Minimization (BM) intractability results).

On the other hand there are subclasses of Boolean functions for which CNF minimization can be done in polynomial time. The easiest case is the class of monotone functions since every monotone function f has a unique prime and irredundant CNF representation (the canonical CNF consisting of all prime implicates of f) which is therefore also the minimum CNF representation. Moreover the canonical CNF of f can be obtained from any monotone CNF representing f in polynomial time simply by performing all possible absorptions. Similarly, every acyclic Horn function f has a unique prime and irredundant CNF which is of course a minimum CNF and can be constructed from an arbitrary Horn CNF of f in polynomial time [15]. Another class that admits polynomial time minimization is the class of quadratic functions [15]. In this case the preprocessing step that outputs an irredundant and prime CNF representation of the input function is not sufficient. It can be shown that the minimization of a quadratic function can be reformulated as a problem on a directed graph, namely a problem of finding its transitive reduction, which is known to be solvable in polynomial time [19]. Finally, the results for acyclic Horn functions and quadratic functions can be combined to get a polynomial time minimization algorithm for the class of quasi-acyclic Horn function [15].

There are several known lower bounds for the size of the minimum CNF representation. For instance, the number of essential prime implicates provides such a lower bound (a prime implicate is essential if it appears in every prime CNF representation of the given function). The notion of an essential implicate was generalized to the notion of an essential set of implicates in [4]. It was proved there, that the maximum number of pairwise disjoint essential sets constitutes a lower bound for the number of clauses in a minimum CNF. Moreover this bound was proved to be tight for the classes of monotone, quadratic, acyclic Horn and quasi-acyclic Horn functions.

Recently, a class of CQ Horn functions was introduced [5]. This class properly contains negative, quadratic, acyclic Horn, and quasi-acyclic Horn functions. The main result in [5] is a polynomial time minimization algorithm for this new class. In this paper we shall show that the lower bound presented in [4] (i.e., the maximum number of pairwise disjoint essential sets) is a tight lower bound on the size of the minimum CNF for the class of CQ Horn functions. In other words, we shall prove that given a CQ Horn function f, the maximum number of pairwise disjoint essential sets of implicates of f equals the minimum number of clauses in a CNF representation of f. Functions which satisfy this equality are called coverable in [8], and so, using this terminology, the main result of this paper can be simply reformulated as stating that every CQ Horn function is coverable.

2 Basic notation, definitions, and results

In this section we shall introduce the necessary notation and summarize the basic known results that will be needed later in the text.

2.1 Boolean functions

A *Boolean function* f on n propositional variables $x_1, \dots, x_n$ is a mapping $\{0,1\}^n \rightarrow \{0,1\}$. The propositional variables $x_1, \dots, x_n$ and their negations $\overline{x}_1, \dots, \overline{x}_n$ are called *literals* (*positive* and *negative literal*s, respectively). An elementary disjunction of literals is called a *clause*, if every propositional variable or its negation appears in it at most once. A clause C is called an *implicate* of a function f if for every $x \in \{0,1\}^n$ we have $f(x) \le C(x)$ (i.e., if f evaluates to 1 on the vector x then C must also evaluate to 1 on x). An implicate C is called *prime* if dropping any literal from it produces a clause which is not an implicate.

It is a well-known fact that every Boolean function f can be represented by a conjunction of clauses (see e.g. [11]). Such an expression is called a *conjunctive normal form* (or CNF) of the Boolean function f. In the rest of the paper we shall often identify a CNF ϕ with a set of its clauses and we shall use both notions interchangeably. A CNF ϕ representing a function f is called *prime* if each clause of ϕ is a prime implicate of the function f. A CNF ϕ representing a function f is called *irredundant* if dropping any clause from ϕ produces a CNF that does not represent f.

Two clauses C_1 and C_2 are said to be *resolvable* if they contain exactly one complementary pair of literals. That means that we can write $C_1 = \tilde{C}_1 \vee x$ and $C_2 = \tilde{C}_2 \vee \overline{x}$ for some propositional variable x and clauses $\tilde{C}_1$ and $\tilde{C}_2$ which contain no complementary pair of literals. The clauses C_1 and C_2 are called *parent clauses* and the disjunction $R(C_1, C_2) = \tilde{C}_1 \vee \tilde{C}_2$ is called the *resolvent* of the parent clauses C_1 and C_2. Note that the resolvent is a clause (does not contain a propositional variable and its negation). It is a well known fact (see [6]), that a resolvent of two implicates of f is an implicate of f and every prime implicate of f can be obtained by a series of resolutions from any CNF representing f (see [6, 17]). So called Quine's procedure takes a CNF φ as an input and it outputs the list of all prime implicates of function represented by φ, reader can find description of this procedure in [6, 17].

For a Boolean function f let us denote by $\mathcal{I}^p(f)$ the set of its prime implicates, and let $\mathcal{I}(f)$ denote the set of implicates, which can be obtained by a series of resolutions from prime implicates (i.e., the resolution closure of $\mathcal{I}^p(f)$).

A clause C is called *negative* if it contains no positive literals. It is called *pure Horn* if it contains exactly one positive literal. To simplify notation, we shall sometimes write a pure Horn clause $C = \bigvee_{x \in S} \overline{x} \vee y$ simply as $C = S \vee y$. Each propositional variable $x \in S$ is called a *subgoal of* C and the propositional variable y is called the *head of* C. We shall denote $Head(C) = y$, $Subg(C) = S$, and $Vars(C) = S \cup \{y\}$.

A CNF is called *Horn* if it contains only negative and pure Horn clauses. A CNF is called *pure Horn* if it contains only pure Horn clauses. Finally, a Boolean function is called *Horn* if it has at least one representation by a Horn CNF, and similarly a Boolean function is called *pure Horn* if it has at least one representation by a pure Horn CNF.

It is known (see [12]) that each prime implicate of a Horn function is either negative or pure Horn, and each prime implicate of a pure Horn function is pure Horn. Thus, in particular, any prime CNF representing a Horn function is Horn, and any prime CNF representing a pure Horn function is pure Horn.

2.2 Forward chaining procedure

In verifying that a given clause is an implicate of a given pure Horn function, a very useful and simple procedure is the following. Let η be a pure Horn CNF of a pure Horn function h. We shall define a *forward chaining* procedure [15] which associates to any subset Q of the propositional variables of h a set M in the following way. The procedure takes as input the subset Q of propositional variables, initializes the set $M = Q$, and at each step it looks for a pure Horn clause $S \vee y$ in η such that $S \subseteq M$, and $y \notin M$. If such a clause is found, the propositional variable y is included into M, and the search is repeated as many times as possible. The set M output by this procedure will be denoted by $FC_\eta(Q)$, where η is the input CNF and Q the starting set of variables. It can be shown [13, 18], that a clause $C = Q \vee y$ is an implicate of h if and only if $y \in FC_\eta(Q)$. If η' and η'' are two distinct CNF representations of a given pure Horn function h and if Q is an arbitrary subset of the propositional variables, then $FC_{\eta'}(Q) = FC_{\eta''}(Q)$ because η' and η'' have the same set of implicates. Therefore, the set of propositional variables reachable from Q by forward chaining depends only on the underlying function rather than on a particular CNF representation. For this reason, we shall also use the expression $FC_h(Q)$ instead of $FC_\eta(Q)$ whenever we do not want to refer to a specific CNF.

2.3 Implication graphs of Horn functions

Let us recall some very useful definitions from [15], associating directed graphs to Horn CNFs and Horn functions.

Definition 2.1 For a Horn CNF ϕ let $G_\phi = (N, A_\phi)$ be the digraph defined by

$$N = \{x \mid x \text{ is a propositional variable in } \phi\}$$

$$A_\phi = \{(x, y) \mid \exists \text{ a clause } C \in \mathcal{C}(\phi) \text{ which contains both literals } \overline{x} \text{ and } y\}.$$

In other words, for each pure Horn clause C in ϕ, the graph G_ϕ contains as many arcs as is the number of subgoals in C, with each arc going from the corresponding subgoal to the head of C. Since a Horn function can be represented by several different Horn CNFs, we can associate in this way several different graphs to a Horn function. It was shown in [3] that given two prime CNFs ϕ_1 and ϕ_2 both representing the same function f, G_{ϕ_1} and G_{ϕ_2} have the same transitive closure which will be denoted by G_f. In what follows we shall call G_ϕ and G_f the *implication graph*s of ϕ and f, respectively. In [5] it was shown that if f is a pure Horn function and $C \in \mathcal{I}(h)$, then $(x, Head(C))$ is an arc of G_f for every $x \in Subg(C)$.

The following graph-theoretic notion will often be used in the rest of paper. Let G be a directed graph and let v be one of its vertices, then $Cone_G(v)$ denotes the set of vertices, where a vertex $u \in Cone_G(v)$ if and only if there is a directed path from u to v in G.

2.4 CQ functions

CQ Horn functions were defined in [5]. Their definition is based on implication graphs and it generalizes the definitions of acyclic and quasi-acyclic functions [15].

Definition 2.2 Let us call a pure Horn clause C *component-wise quadratic* (or CQ for short) with respect to a Horn function f if at most one subgoal of C belongs to the strong component of G_f containing the head of C. A Horn CNF ϕ representing a function f is said to be CQ if every pure Horn clause of ϕ is CQ with respect to f. Finally, a Horn function f is called CQ if it admits at least one CQ CNF representation.

The following important property of CQ functions was shown in [5].

Theorem 2.3 *Let f be a CQ function and C an arbitrary clause in $\mathcal{I}(f)$. Then C is CQ with respect to f.*

2.5 Essential sets of implicates

In this section we shall recall the notion of essential set introduced in [4] and some of their properties proved there. In the remainder of this section let us consider an arbitrary but fixed Boolean function f.

Definition 2.4 Given a set $\mathcal{C}$ of clauses, a subset $\mathcal{E} \subseteq \mathcal{C}$ is called an *essential subset of* $\mathcal{C}$ if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{C}$ the following implication holds:

$$R(C_1, C_2) \in \mathcal{E} \Longrightarrow C_1 \in \mathcal{E} \text{ or } C_2 \in \mathcal{E},$$

i.e., the resolvent belongs to $\mathcal{E}$ only if at least one of the parent clauses are from $\mathcal{E}$. In particular, if $\mathcal{C} = \mathcal{I}(f)$ for a Boolean function f, we call $\mathcal{E}$ an essential set of clauses of f (or simply an essential set, if f or $\mathcal{C}$ is clear from the context).

The orthogonality property of essential sets was proved in [4]. This key proposition (which appears there as Theorem 6.4) shows that every essential set has one (or more) of its clauses present in every representation of f and moreover that this condition is not only necessary but also sufficient.

Proposition 2.5 [4] *Let $\mathcal{C} \subseteq \mathcal{I}(f)$ be arbitrary. Then $\mathcal{C}$ represents f if and only if $\mathcal{C} \cap \mathcal{E} \neq \emptyset$ for every nonempty essential set of clauses $\mathcal{E} \subseteq \mathcal{I}(f)$.*

Proposition 2.5 has an obvious corollary: if there exist nonempty essential sets $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_k \subseteq \mathcal{I}(f)$ which are pairwise disjoint, then every representation of f must consist of at least k clauses. Hence, any collection of pairwise disjoint essential sets of clauses provides an easy lower bound on the size (i.e., number of clauses) of a minimal representation of f. The authors posed a question in [4] asking for which classes of Boolean functions this lower bound is in fact tight. They also gave a partial answer to this question by proving, that the bound is tight for the classes of monotone, quadratic, acyclic and quasi-acyclic functions.

It was shown in [15] that every prime and irredundant CNF representation of a Horn function f contains the same number of negative clauses. This result was later extended in [4] by showing that this is also the maximum number of pairwise disjoint essential sets of negative implicates of f. Following this result, it was also observed in [4] that to find pairwise disjoint essential sets of implicates of a Horn function f, it is enough to find pairwise disjoint essential sets of a pure Horn function f' represented by pure Horn implicates of f, and then to add pairwise disjoint essential sets of negative implicates of f. In the rest of the paper we shall therefore restrict our attention only to pure Horn functions.

The following essential sets will play important role in the proof of our result.

Definition 2.6 Given a Boolean function f on n propositional variables and an arbitrary vector $t \in \{0, 1\}^n$ let us define a *falsepoint set of f defined by t* as

$$\mathcal{E}(t) = \{C \in \mathcal{I}(f) \mid C(t) = 0\}$$

where by $C(t) = 0$ we mean the following : if we substitute for the propositional variables of f the truth values according to the vector t then the clause C evaluates to zero (false).

It was shown in [4], that the sets defined in Definition 2.6 are essential sets of implicates of f.

We shall show, that given a pure Horn CQ function f with the number of clauses in a shortest CNF representing f equal to k, we can find falsepoints $t_1, \dots, t_k$, such that for every two different falsepoints t_i and t_j we have $\mathcal{E}(t_i) \cap \mathcal{E}(t_j) = \emptyset$. This will show, that the maximum number of pairwise disjoint essential sets of f is equal to the size of the shortest CNF representing f.

2.6 Clause graphs

For a definition of suitable falsepoints we need to recall the definition of yet another directed graph, associated to a set of pure Horn clauses and/or to a pure Horn function. As opposed to G_ϕ and G_f which are defined on the set of variables, this so called *clause graph* [5] is defined on the set of clauses.

Given a set $\mathcal{C}$ of pure Horn clauses let us define its *clause graph* $D_\mathcal{C} = (V_\mathcal{C}, E_\mathcal{C})$. The vertex set of the clause graph $V_\mathcal{C} = V(D_\mathcal{C}) = \mathcal{C}$ is the given set of clauses, and the edge set $E_\mathcal{C} = E(D_\mathcal{C})$ is defined as follows: For $C_1, C_2 \in \mathcal{C}$ we have $(C_1, C_2) \in E_\mathcal{C}$ if and only if both

(1) $Head(C_1) \in Cone_{G_\mathcal{C}}(Head(C_2))$, and
(2) $Subg(C_1) \subseteq FC_\mathcal{C}(Subg(C_2))$.

It is easy to see by the definitions of the implication graph $G_\mathcal{C}$ and of the forward chaining procedure that the clause graph $D_\mathcal{C}$ is transitively closed. In the special case when $\mathcal{C} = \mathcal{I}(f)$ for a pure Horn function f we shall denote the clause graph of $\mathcal{C}$ by D_f. Furthermore, whenever the function f will be clear from the context, we shall simply write D instead of D_f. In this latter case condition (1) simplifies to $(Head(C_1), Head(C_2)) \in A_f$, due to the fact that $G_f = (N, A_f)$ is itself transitively closed.

Let f be a CQ pure Horn function and let $C, C' \in \mathcal{I}(f)$ be two implicates of f, which belong to the same strong component of D_f. Let Q (Q' resp.) be a strong component of G_f which contains $Head(C)$ ($Head(C')$ resp.). It was shown in [5] that

$|Subg(C) \cap Q| = |Subg(C') \cap Q'|$. In particular, either both intersections are empty or both have cardinality one.

3 Disjoint essential sets for the class of CQ functions

In this section we shall show the main result of this paper, which is contained in the following theorem.

Theorem 3.1 *Let f be a CQ pure Horn function on n variables and let $\varphi = \bigwedge_{i=1}^{k} C_i$ be a CNF representing f which has the least number of clauses. Then there exist falsepoints $t_1, \dots, t_k$ of f, such that the corresponding essential sets $\mathcal{E}(t_1), \dots, \mathcal{E}(t_k)$ are pairwise disjoint.*

Proof of Theorem 3.1 is contained in the rest of this section. We shall suppose, that φ is the shortest CNF representing f which has been found by the algorithm for minimizing CQ formulae described in [5]. We refer the reader to [5] for details on the structure of CNF φ.

In the rest of this section, we shall use the following notation: G denotes an implication graph of f, D_φ denotes a clause graph of φ, D denotes a clause graph of f. Given a variable z, $Q(z)$ denotes the strong component of G which contains z. Given an implicate $C \in \mathcal{I}(f)$, $K(C)$ denotes the strong component of D, which contains C.

For every $i \in \{1, \dots, k\}$ let $C_i = A_i \vee X_i \vee u_i$, where $A_i \cap Q(u_i) = \emptyset$, $X_i \subseteq Q(u_i)$, and $|X_i| \leq 1$. Let z be an arbitrary variable, let $\varphi_i = Cone_{D_\varphi}(C_i)$, and let $Q_i(z)$ denote the strong component of G_{φ_i}, which contains z. Note that clearly $Q_i(z) \subseteq Q(z)$ and it may happen that $Q_i(z) \subset Q(z)$ (i.e., G_{φ_i} may have several strong components inside a single strong component of G) because φ_i contains less clauses than φ. It can be shown, that φ_i represents the same function as $Cone_D(C_i)$, see [4]. Similarly, $\varphi_i \setminus K(C_i)$ represents the same function as $Cone_D(C_i) \setminus K(C_i)$. In particular, let z be a variable and B a set of variables, then it can be shown, that if $(B \vee z) \in Cone_D(C_i)$, then $z \in FC_{\varphi_i}(B) = FC_{Cone_D(C_i)}(B)$, for more details, see [4, 5]. Let $t_i \in \{0, 1\}^n$ be a vector, which is defined as follows.

Let us at first assume, that $X_i \cap Q_i(u_i) = \emptyset$ (this includes both the case when $X_i = \emptyset$ and the case when $X_i = \{x_i\}$ for some variable $x_i \in Q(u_i) \setminus Q_i(u_i)$).

$$t_i[z] = \begin{cases} 1 & \text{if } z \notin Cone_G(u_i) \\ 1 & \text{if } z \in FC_{\varphi_i \setminus K(C_i)}(A_i \cup X_i) \\ 0 & \text{if } z \in Cone_G(u_i) \setminus Q(u_i) \text{ and } z \notin FC_{\varphi_i \setminus K(C_i)}(A_i \cup X_i) \\ 1 & \text{if } z \in Q(u_i) \setminus Q_i(u_i) \text{ and } z \in FC_{\varphi_i}(A_i \cup X_i) \\ 0 & \text{if } z \in Q(u_i) \setminus Q_i(u_i) \text{ and } z \notin FC_{\varphi_i}(A_i \cup X_i) \\ 0 & \text{if } z \in Q_i(u_i) \text{ and } z \notin FC_{\varphi_i \setminus K(C_i)}(A_i \cup X_i) \end{cases}$$

For a more illustrative definition of vector t_i, see Fig. 1.

Now, we shall assume, that $X_i = \{x_i\}$ for some variable $x_i \in Q_i(u_i)$. Let ms_i denote the number of source strong components of graph $G_{\varphi_i \setminus K(C_i)}$ within $Q_i(u_i)$ and let mt_i

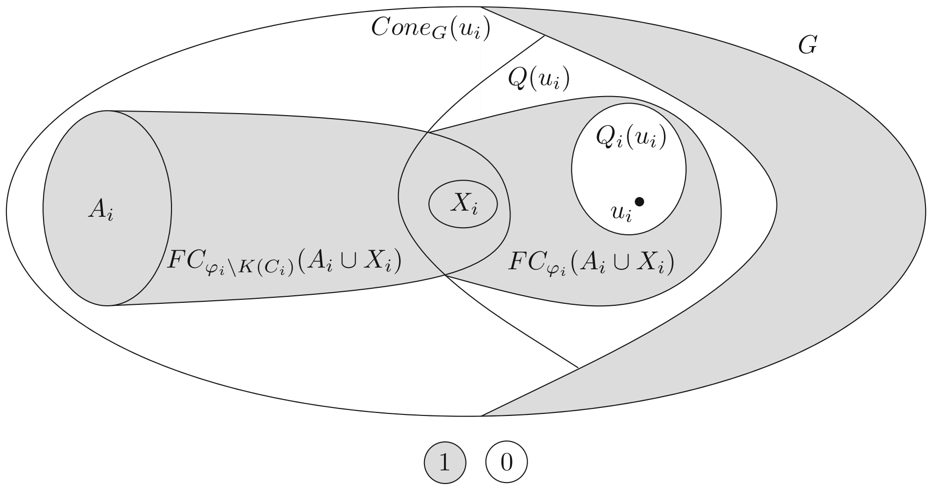
$Cone_G(u_i)$
G
$Q(u_i)$
$Q_i(u_i)$
A_i
X_i
u_i
$FC_{\varphi_i \setminus K(C_i)}(A_i \cup X_i)$
$FC_{\varphi_i}(A_i \cup X_i)$
1
0

Fig. 1 Definition of vector t_i when $X_i \cap Q_i(u_i) = \emptyset$

denote the number of sink strong components of graph $G_{\varphi_i \setminus K(C_i)}$ within $Q_i(u_i)$ for $i = 1, \dots, k$.

$$t_i[z] = \begin{cases} 1 \text{ if } z \notin Cone_G(u_i) \\ 1 \text{ if } z \in FC_{\varphi_i \setminus K(C_i)}(A_i), \text{ this has precedence before the next rules} \\ 0 \text{ if } z \in Cone_G(u_i) \setminus Q(u_i) \text{ and } z \notin FC_{\varphi_i \setminus K(C_i)}(A_i) \\ 1 \text{ if } z \in Q(u_i) \setminus Q_i(u_i) \text{ and } z \in FC_{\varphi_i}(A_i \cup X_i) \\ 0 \text{ if } z \in Q(u_i) \setminus Q_i(u_i) \text{ and } z \notin FC_{\varphi_i}(A_i \cup X_i) \\ 0 \text{ if } z \in Q_i(u_i) \text{ and } z \text{ is in the same strong component of } G_{\varphi_i \setminus K(C_i)} \text{ as } u_i \\ 1 \text{ if } z \in Q_i(u_i) \text{ and } z \text{ is in the same strong component of } G_{\varphi_i \setminus K(C_i)} \text{ as } x_i \\ 0 \text{ if } ms_i \le mt_i,\ z \in Q_i(u_i), \text{ and} \\ \qquad z \text{ is in a different strong component of } G_{\varphi_i \setminus K(C_i)} \text{ than } x_i \\ 1 \text{ if } ms_i > mt_i,\ z \in Q_i(u_i), \text{ and} \\ \qquad z \text{ is in a different strong component of } G_{\varphi_i \setminus K(C_i)} \text{ than } u_i. \end{cases}$$

For a more illustrative definition of vector t_i, see Fig. 2. Here the white color denotes value 0, darker gray color denotes value 1, and lighter gray color denotes either value 0 if $ms_i \le mt_i$, or value 1 if $ms_i > mt_i$. Circles enclosing X_i and u_i denote the strong components of $G_{\varphi_i \setminus K(C_i)}$ which contain X_i and u_i.

In the rest of this section we may assume, that for every $i = 1, \dots, k$ the following invariants are satisfied.

(I) If $ms_i \le mt_i$, then the strong component of $G_{\varphi_i \setminus K(C_i)}$ containing x_i does not have any outgoing arcs. If $ms_i > mt_i$, then the strong component of $G_{\varphi_i \setminus K(C_i)}$ containing u_i does not have any incoming arcs.

(II) If C_i and C_j belong to $\varphi_i \cap K(C_i)$, and $Q_i(u_i) = Q_i(u_j) = Q_i(x_i) = Q_i(x_j)$, then $i \neq j$ implies $x_i \neq x_j$ and $u_i \neq u_j$.

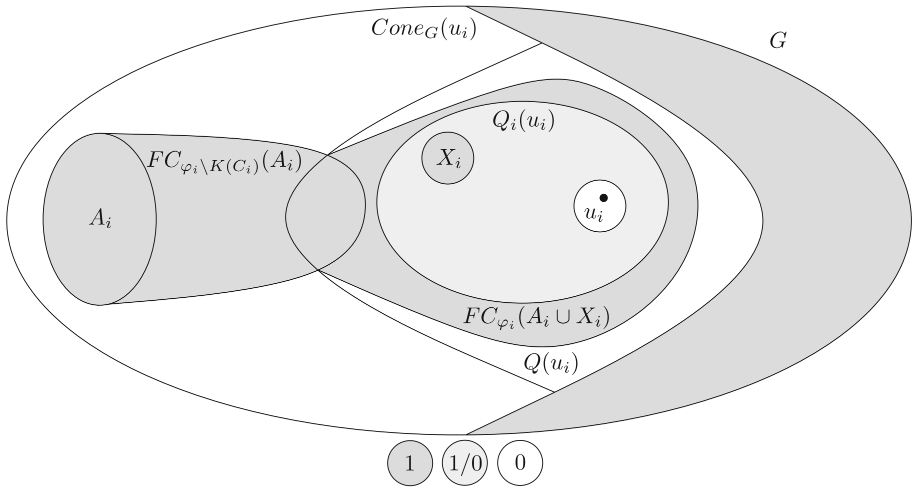
$Cone_G(u_i)$
G
$Q_i(u_i)$
$FC_{\varphi_i\setminus K(C_i)}(A_i)$
X_i
A_i
u_i
$FC_{\varphi_i}(A_i \cup X_i)$
$Q(u_i)$
1
1/0
0

Fig. 2 Definition of vector t_i when $X_i \subseteq Q_i(u_i)$

The correctness of these invariants follows from the properties of the algorithm described in [5] which constructs the CNF φ. We shall now state and prove a sequence of technical lemmas which will show that for every $i = 1, \ldots, k$ we have $C_i \in \mathcal{E}(t_i)$ (implying that t_i is a falsepoint of f and the essential set $\mathcal{E}(t_i)$ is nonempty), and that for every $j \in \{1, \ldots, k\} \setminus \{i\}$ we have $\mathcal{E}(t_i) \cap \mathcal{E}(t_j) \neq \emptyset$ (i.e., that the essential sets are pairwise disjoint).

Lemma 3.2 *For every $i = 1, \ldots, k$ we have $C_i \in \mathcal{E}(t_i)$.*

Proof Let us assume that $X_i \cap Q_i(u_i) = \emptyset$. Clearly, $A_i \cup X_i \subseteq FC_{\varphi_i\setminus K(C_i)}(A_i \cup X_i)$, and therefore for every $z \in A_i \cup X_i$ we have that $t_i[z] = 1$. Also $t_i[u_i] = 0$, because obviously $u_i \in Q_i(u_i)$.

Now, let us assume, that $X_i = \{x_i\}$ for some variable $x_i \in Q_i(u_i)$. Clearly $A_i \subseteq FC_{\varphi_i\setminus K(C_i)}(A_i)$, therefore for every $z \in A_i$ we have $t_i[z] = 1$. On the other hand neither x_i, nor u_i belong to $FC_{\varphi_i\setminus K(C_i)}(A_i)$ because, otherwise C_i would not be a prime implicate. Since $x_i \in Q_i(u_i)$ and x_i belongs to the same strong component of $G_{\varphi_i\setminus K(C_i)}$ as x_i, we have that $t_i[x_i] = 1$ and for similar reasons we have that $t_i[u_i] = 0$.

In both cases $C_i(t_i) = 0$ and hence $C_i \in \mathcal{E}(t_i)$. □

Lemma 3.3 *Let $i \in \{1, \ldots, k\}$ be arbitrary with $X_i \cap Q_i(u_i) = \emptyset$, then $\mathcal{E}(t_i) \subseteq K(C_i)$.*

Proof Let $(Z \vee v) \in \mathcal{E}(t_i)$ be an arbitrary implicate of f. Let us at first show that $(Z \vee v) \in Cone_D(C_i)$. For this we need, that $Z \subseteq FC_{\varphi}(A_i \cup X_i)$, and that $v \in Cone_G(u_i)$. If $v \notin Cone_G(u_i)$, then we would have $t_i[v] = 1$, therefore $v \in Cone_G(u_i)$, and hence $Z \subseteq Cone_G(u_i)$ as well. Now let $z \in Z$ be arbitrary, and let us show, that $z \in FC_{\varphi}(A_i \cup X_i)$. If $z \in FC_{\varphi_i\setminus K(C_i)}(A_i \cup X_i)$, then clearly also $z \in FC_{\varphi}(A_i \cup X_i)$. Now, let us assume that $z \in Q(u_i) \setminus Q_i(u_i)$, but then $z \in FC_{\varphi_i}(A_i \cup X_i)$, and we are done.

Now, let us assume by contradiction, that $(Z \vee v) \in Cone_D(C_i) \setminus K(C_i)$. Since for every $z \in Z$ we have $t_i[z] = 1$ and $z \in Cone_G(u_i)$, we have that $z \in FC_{\varphi_i}(A_i \cup X_i)$, and therefore also $Z \subseteq FC_{\varphi_i}(A_i \cup X_i)$. If $Z \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i \cup X_i)$, then we would have that $v \in FC_{\varphi_i \setminus K(C_i)}(A_i \cup X_i)$, which is not possible, because in this case we would have $t_i[v] = 1$. Therefore there must be a variable $y \in Z$ such that $y \notin FC_{\varphi_i \setminus K(C_i)}(A_i \cup X_i)$. It follows that $y \in Q(u_i) \setminus Q_i(u_i)$. Since f is a CQ function, there can be at most one such variable. Let $B = Z \setminus \{y\}$, so that we can write $(B \vee \overline{y} \vee v) = (Z \vee v)$. Since such a y exists, it follows, that $v \in Q(u_i)$ and hence also $v \in Q_i(u_i)$, because $v \in FC_{\varphi_i}(A_i \cup X_i)$ and $t_i[v] = 0$. This means that there is a path from v to u_i in G_{φ_i}, and this path cannot use the arc generated by C_i, because none of the subgoals of C_i belongs to $Q_i(u_i)$. Therefore there is in fact a path from v to u_i in $G_{\varphi_i \setminus \{C_i\}}$. Since $(B \vee \overline{y} \vee v) \in Cone_D(C_i) \setminus K(C_i)$, we have that there is a path from y to v in $G_{\varphi_i \setminus K(C_i)}$ and hence there is also a path from y to u_i in graph $G_{\varphi_i \setminus \{C_i\}}$. From this we shall derive that $u_i \in FC_{\varphi_i \setminus \{C_i\}}(A_i \cup X_i)$, which will yield a contradiction to irredundancy of φ_i. Let $D_1, \ldots, D_\ell \in \varphi_i$ be an irredundant forward chaining derivation of y from $A_i \cup X_i$. If a clause C_i would be present in this derivation, it would mean that $y \in Q_i(u_i)$, therefore C_i is not present in this derivation. Now, let $(E_1 \vee \overline{y} \vee w_1), (E_2 \vee \overline{w}_1 \vee w_2), \ldots, (E_m \vee \overline{w}_{m-1} \vee u_i) \in \varphi_i \setminus \{C_i\}$ be clauses, which generate the path from y to u_i in graph $G_{\varphi_i \setminus \{C_i\}}$. For each $j = 1, \ldots, m$ we clearly have that $E_j \subseteq FC_{\varphi_i}(A_i \cup X_i)$ because all clauses in $\varphi_i \setminus \{C_i\}$ belong to $Cone_D(C_i)$, which means that $E_j \subseteq FC_\varphi(A_i)$. However, every clause used in the forward chaining derivation of variables in E_j from A_i is in φ_i, because $E_j \subseteq Cone_G(u_i)$. Moreover, since $E_j \cap Q(u_i)$ is necessarily empty, we have that in fact $E_j \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i)$. Therefore we get that $E_1 \cup \{y\} \subseteq FC_{\varphi_i \setminus \{C_i\}}(A_i \cup X_i)$. Since $(E_1 \vee \overline{y} \vee w_1) \in \varphi_i \setminus \{C_i\}$, we have, that also $w_1 \in FC_{\varphi_i \setminus \{C_i\}}$. By a simple induction we can derive that in fact for every $j = 1, \ldots, m-1$ we have $w_j \in FC_{\varphi_i \setminus \{C_i\}}(A_i \cup X_i)$ and in particular that $u_i \in FC_{\varphi_i \setminus \{C_i\}}(A_i \cup X_i)$. However, this cannot happen because of the irredundancy of φ, and hence we have shown that $(Z \vee v) = (B \vee \overline{y} \vee v)$ belongs to $K(C_i)$. □

Lemma 3.4 *Let $i \in \{1, \ldots, k\}$ be an arbitrary index with $X_i = \{x_i\}$ for some variable $x_i \in Q_i(u_i)$. Then $\mathcal{E}(t_i) \subseteq K(C_i)$. Moreover, if a clause $(B \vee \overline{y} \vee v) \in \mathcal{E}(t_i)$, where $B \cap Q(u_i) = \emptyset$ and $y, v \in Q(u_i)$, then $y, v \in Q_i(u_i)$.*

Proof Let $(Z \vee v) \in \mathcal{E}(t_i)$ be an arbitrary implicate.

Let us at first show, that $(Z \vee v) \in Cone_D(C_i)$. For this we need, that $Z \subseteq FC_\varphi(A_i \cup X_i)$ and that $v \in Cone_G(u_i)$. We can observe, that if $v \notin Cone_G(u_i)$, then we would have $t_i[v] = 1$, hence $v \in Cone_G(u_i)$ and $Z \subseteq Cone_G(u_i)$ as well. Let $z \in Z$ be arbitrary and let us show that $z \in FC_\varphi(A_i \cup X_i)$. Since $t_i[z] = 1$ and $z \in Cone_G(u_i)$, we have to distinguish three cases.

- $z \in FC_{\varphi_i \setminus K(C_i)}(A_i)$. In this case trivially $z \in FC_\varphi(A_i)$.
- $z \in Q_i(u_i)$. The fact that $z \in Q_i(u_i)$ implies, that there is a path from z to u_i in G_{φ_i}, let z' be a variable, such that $(z, z') \in G_{\varphi_i}$. Therefore there must be a clause $(V \vee \overline{z} \vee z') \in \varphi_i$. In particular, $(V \vee \overline{z} \vee z') \in Cone_{D_\varphi}(C_i)$, hence $z \in FC_\varphi(A_i \cup X_i)$.
- $z \in Q(u_i) \setminus Q_i(u_i)$, and $z \in FC_{\varphi_i}(A_i \cup X_i)$, in this case trivially $z \in FC_\varphi(A_i \cup X_i)$.

Now let us show, that $(Z \vee v) \in K(C_i)$. We shall proceed by contradiction and assume, that $(Z \vee v) \in Cone_D(C_i) \setminus K(C_i)$. Assume for a while, that $Z \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i)$. Then also $v \in FC_{\varphi_i \setminus K(C_i)}(A_i)$, which would mean, that $t_i[v] = 1$. Therefore there is a variable $y \in Z$ for which $y \notin FC_{\varphi_i \setminus K(C_i)}(A_i)$. Since $t_i[y] = 1$ and $y \in Cone_G(u_i)$, we have that $y \in Q(u_i)$. Together with the fact that $v \in Cone_G(u_i)$ we also get that $v \in Q(u_i)$. Since f is a CQ function, there can be at most one such variable y. Let us denote by $B = Z \setminus \{y\}$, so that we can write $(Z \vee v) = (B \vee \overline{y} \vee v)$, where $B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i)$.

Let us at first show, that $z \in Q_i(u_i)$. The facts that $y \in Q(u_i)$ and $t_i[y] = 1$ together imply $y \in FC_{\varphi_i}(A_i \cup X_i)$. Therefore $v \in FC_{\varphi_i}(A_i \cup X_i)$ as well (recall that $B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i)$ holds) and thus $t_i[v] = 0$ forces $v \in Q_i(u_i)$. Now let us show, that $y \in Q_i(u_i)$. Assume by contradiction that $y \in Q(u_i) \setminus Q_i(u_i)$. From $(B \vee \overline{y} \vee v) \in Cone_D(C_i)$ and $v \in Q_i(u_i)$ it follows that there is a path from y to u_i in G_{φ_i}. In fact, we can even observe, that $y \in FC_{\varphi_i}(A_i \cup X_i)$ can be strengthened to $y \in FC_{\varphi_i}(A_i)$, since otherwise there would be a path from x_i to y in G_{φ_i} implying $y \in Q_i(u_i)$ and contradicting our assumption. Hence also $v \in FC_{\varphi_i}(A_i)$. We can even conclude now that $v \in FC_{\varphi_i \setminus K(C_i)}(A_i)$, since otherwise, let $(A_j \vee x_j \vee u_j) \in K(C_i)$ be an arbitrary clause used in an irredundant forward chaining derivation of v from A_i. Then $A_i \subseteq FC_{\varphi_i}(A_j)$ and $x_j \in FC_{\varphi_i}(A_i)$, and therefore $(A_j \vee u_j)$ is an implicate of f contradicting the primality of φ_i. The fact that $v \in FC_{\varphi_i \setminus K(C_i)}(A_i)$ however implies, that $t_i[v] = 1$, and that is a contradiction as well. Thereore $y \in Q_i(u_i)$ must hold.

Note, that when deriving $y, z \in Q_i(u_i)$ in the previous paragraph, we have not used the fact, that $(Z \vee v) = (B \vee \overline{y} \vee v) \in K(C_i)$. We have only used the previously derived fact that $y, z \in Q(u_i)$. Therefore $y, z \in Q_i(u_i)$ holds for every implicate $(B \vee \overline{y} \vee v) \in \mathcal{E}(t_i)$ for which $B \cap Q(u_i) = \emptyset$ and $y, v \in Q(u_i)$, which finishes the proof of the second statement of the lemma.

To complete the proof of the first statement of the lemma let us distinguish two cases according to which of ms_i and mt_i is bigger.

- If $ms_i \leq mt_i$ then let us denote by Q the strong component of $G_{\varphi_i \setminus K(C_i)}$ containing x_i. In this case $t_i[y] = 1$ implies $y \in Q$ and $t_i[v] = 0$ implies $v \notin Q$. Since according to invariant (I) there is no arc going out of Q, it is not possible that $(B \vee \overline{y} \vee v) \in Cone_D(C_i) \setminus K(C_i)$.
- If $ms_i > mt_i$ then let us denote by Q the strong component of $G_{\varphi_i \setminus K(C_i)}$ containing u_i. In this case $t_i[y] = 1$ implies $y \notin Q$ and $t_i[v] = 0$ implies $v \in Q$. Since according to invariant (I) there is no arc going into Q, it is not possible that $(B \vee \overline{y} \vee v) \in Cone_D(C_i) \setminus K(C_i)$.

Since the assumption that $(B \vee \overline{y} \vee v) \in Cone_D(C_i) \setminus K(C_i)$ leads to a contradiction in both cases, we can conclude, that $(B \vee \overline{y} \vee v) \in K(C_i)$. □

Now, let us show, that given $C_i, C_j \in \varphi$ with $i \neq j$ we have $\mathcal{E}(t_i) \cap \mathcal{E}(t_j) = \emptyset$. Since this trivially holds when $K(C_i) \neq K(C_j)$, we need to consider only the case when $K(C_i) = K(C_j)$.

Lemma 3.5 *Let $C_i = (A_i \vee X_i \vee u_i)$, $C_j = (A_j \vee X_j \vee u_j) \in \varphi$ where $i \neq j$, and $X_i = X_j = \emptyset$ be two clauses such that $K(C_i) = K(C_j)$. Then $\mathcal{E}(t_i) \cap \mathcal{E}(t_j) = \emptyset$.*

Proof Let us proceed by contradiction and let us consider an implicate $(B \vee v) \in \mathcal{E}(t_i) \cap \mathcal{E}(t_j)$. Using Lemma 3.3 we get $(B \vee v) \in K(C_i) = K(C_j)$, which implies $v \in Q(u_i) = Q(u_j)$, and that in turn implies $B \subseteq Cone_G(u_i) = Cone_G(u_j)$. Since for every $z \in B$ we have $t_i[z] = t_j[z] = 1$, we have that $B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i)$ and

$B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_j)$. Since $(B \vee v) \in K(C_i)$, we have that $v \in FC_{\varphi_i}(A_i)$ and $v \in FC_{\varphi_i}(A_j)$, and since $t_i[v] = t_j[v] = 0$, we also have that $v \in Q_i(u_i)$ and $v \in Q_j(u_j)$. In particular this implies that $Q_i(u_i) = Q_j(u_j)$. Since $A_j \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i)$, we must have that $j = i$, otherwise we would get a contradiction with irredundancy of φ, because we could generate $(A_i \vee u_i)$ from $\varphi \setminus \{(A_i \vee u_i)\}$. Therefore given $i \neq j$, we must have that $\mathcal{E}(t_i) \cap \mathcal{E}(t_j) = \emptyset$. □

Lemma 3.6 *Let* $C_i = (A_i \vee \overline{x}_i \vee u_i), C_j = (A_j \vee \overline{x}_j \vee u_j) \in \varphi$ *where* $i \neq j$ *be two clauses such that* $K(C_i) = K(C_j)$, $x_i \in Q(u_i) \setminus Q_i(u_i)$, *and* $x_j \in Q(u_j) = Q(u_i)$. *Then* $\mathcal{E}(t_i) \cap \mathcal{E}(t_j) = \emptyset$.

Proof Let us proceed by contradiction and let us consider an implicate $(B \vee y \vee v) \in \mathcal{E}(t_i) \cap \mathcal{E}(t_j)$, where $B \cap Q(u_i) = \emptyset$, $y, v \in Q(u_i) = Q(u_j)$.

Let us at first consider the case when also $x_j \in Q(u_j) \setminus Q_j(j)$. Since for every $z \in B$ we have $t_i[z] = t_j[z] = 1$, we have that $B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i \cup \{x_i\})$ and $B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_j \cup \{x_j\})$, in fact since if x_i (x_j resp.) would be necessary in this forward chaining derivation, we would have that some element of B belongs to $Q(u_i)$, therefore we have that $B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i)$ and $B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_j)$. Similarly $y \in FC_{\varphi_i}(A_i \cup \{x_i\})$ and $y \in FC_{\varphi_i}(A_j \cup \{x_j\})$. Since $(B \vee y \vee v) \in K(C_i)$, we have that $v \in FC_{\varphi_i}(A_i \cup \{x_i\})$ and $v \in FC_{\varphi_i}(A_j \cup \{x_j\})$, and since $t_i[v] = t_j[v] = 0$, we also have that $v \in Q_i(u_i)$ and $v \in Q_j(u_j)$. In particular this implies that $Q_i(u_i) = Q_i(u_i) = Q_j(u_j)$. We can observe that $A_j \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i)$, because $C_j \in K(C_i)$. Also $x_j \in FC_{\varphi_i \setminus \{C_i\}}(A_i \cup \{x_i\})$, otherwise there would be a path from u_i to x_j in graph G_{φ_i} and therefore x_j would belong to $Q_i(u_i) = Q_j(u_j)$. This together with fact that $Q_i(u_i) = Q_j(u_j)$ implies, that $j = i$, otherwise we would get a contradiction with irredundancy of φ, because we could generate $(A_i \vee \overline{x}_i \vee u_i)$ from $\varphi \setminus \{(A_i \vee \overline{x}_i \vee u_i)\}$. Therefore given $i \neq j$, we must have that $\mathcal{E}(t_i) \cap \mathcal{E}(t_j) = \emptyset$.

Now let us assume, that $x_j \in Q_j(u_j)$. Still $B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_i)$, $y \in FC_{\varphi_i}(A_i \cup \{x_i\})$, $v \in FC_{\varphi_i}(A_i \cup \{x_i\})$, and $v \in Q_i(u_i)$. Let $z \in B$ be arbitrary, since $t_j[z] = 1$, we must have that $z \in FC_{\varphi_i \setminus K(C_i)}(A_j)$ and hence $B \subseteq FC_{\varphi_i \setminus K(C_i)}(A_j)$. According to Lemma 3.4, we also have that $y, v \in Q_j(u_j)$, in particular, we get, that $Q_i(u_i) = Q_i(u_j) = Q_j(u_j)$. We know, that $A_i \subseteq FC_{\varphi_i \setminus K(C_i)}(A_j)$, and $x_i \in FC_{\varphi_i}(A_j \cup \{x_j\})$. If x_j appears in any irredundant forward chaining derivation of x_i from $A_j \cup \{x_j\}$, we would have a path from x_j to x_i in G_{φ_i} and hence we would have that $x_i \in Q_i(u_i) = Q_i(u_j)$. Therefore in fact $x_i \in FC_{\varphi_i}(A_j)$ and hence no clause from $\varphi_i \cap K(C_i)$ can be used in this derivation, assuming φ_i is prime and irredundant. Therefore we get in fact $A_i \cup \{x_i\} \subseteq FC_{\varphi_i \setminus K(C_i)}(A_j)$. This implies that $u_i \in FC_{\varphi_i \setminus \{C_j\}}(A_j)$, together with fact that $Q_i(u_i) = Q_j(u_j)$ we get a contradiction with irredundancy of φ_i since we would be able to derive C_j from $\varphi_i \setminus \{C_j\}$. □

Lemma 3.7 *Let* $C_i = (A_i \vee X_i \vee u_i), C_j = (A_j \vee X_j \vee u_j) \in \varphi$ *where* $i \neq j$ *be two clauses such that* $X_i = \{x_i\}$ *and* $X_j = \{x_j\}$ *for some variables* $x_i \in Q_i(u_i)$ *and* $x_j \in Q_j(u_j)$ *and* $K(C_i) = K(C_j)$. *Then* $\mathcal{E}(t_i) \cap \mathcal{E}(t_j) = \emptyset$.

Proof Let us proceed by contradiction and let us consider an implicate $(B \vee y \vee v) \in \mathcal{E}(t_i) \cap \mathcal{E}(t_j)$, where $B \subseteq FC_{\varphi}(A_i)$, $B \subseteq FC_{\varphi}(A_j)$, $v \in Q(u_i) = Q(u_j)$, $y \in FC_{\varphi}(A_i \cup X_i)$, $y \in FC_{\varphi}(A_j \cup X_j)$, $y \in Q(u_i) = Q(u_j)$. From Lemma 3.4 we also know, that both y and v must belong to $Q_i(u_i)$ and $Q_j(u_j)$, therefore in fact $Q_i(u_i) = Q_i(u_j) = Q_j(u_j)$.

Let us at first assume, that $ms_i \leq mt_i$, then necessarily y belongs to the same strong component of $G_{\varphi_i \setminus K(C_i)}$ as both x_i and x_j, which would imply according to invariant (II) that $i = j$ which is a contradiction.

Similarly if $ms_i > mt_i$, then necessarily v belongs to the same strong component of $G_{\varphi_i \setminus K(C_i)}$ as both u_i and u_j, which would imply according to invariant (II) that $i = j$ which is a contradiction. □

Now we are ready to complete the proof of Theorem 3.1.

Proof Lemma 3.2 shows $\forall 1 \leq i \leq k : C_i \in \mathcal{E}(t_i)$ and thus the vectors t_i are falsepoints of f which define nonempty essential sets $\mathcal{E}(t_i)$ of f. It remains to show that the sets $\mathcal{E}(t_1), \ldots, \mathcal{E}(t_k)$ are pairwise disjoint. Lemmas 3.3 and 3.4 show that $\forall 1 \leq i \leq k : \mathcal{E}(t_i) \subseteq K(C_i)$, i.e., each of the k essential sets sits inside a single strong component of the clause graph D. Hence any two $\mathcal{E}(t_i), \mathcal{E}(t_j)$ contained in two different strong components of D are disjoint.

Lemmas 3.5, 3.6, and 3.7 deal with the case when $\mathcal{E}(t_i), \mathcal{E}(t_j)$ are both inside the same strong component $K(C_i) = K(C_j)$ of the clause graph D. Lemma 3.5 tackles the case when no subgaol of C_i and C_j is in the same strong component $Q(u_i) = Q(u_j)$ of the graph G as their heads. Lemma 3.6 treats the case when both C_i and C_j have exactly one subgoal each (let us call these subgoals x_i and x_j) inside the strong component $Q(u_i) = Q(u_j)$ and moreover $x_i \notin Q_i(u_i)$ or $x_j \notin Q_j(u_j)$ (actually $x_i \notin Q_i(u_i)$ is assumed without a loss of generality). Finally, Lemma 3.7 deals with the case which is similar to the one of Lemma 3.6, except that here both $x_i \in Q_i(u_i)$ and $x_j \in Q_j(u_j)$ hold. Since this covers all possible cases the proof is finished. □

Acknowledgements The authors gratefully acknowledge the support by the Czech Science Foundation (grants 201/07/P168 and P202/10/1188). They would also like to thank the anonymous referee for the valuable comments which helped to improve the presentation of this paper.

References

1. Ausiello, G., D'Atri, A., Sacca, D.: Minimal representation of directed hypergraphs. SIAM J. Comput. **15**(2), 418–431 (1986)
2. Boros, E., Čepek, O.: On the complexity of Horn minimization. Tech. Rep. RUTCOR Research Report RRR 1-1994, Rutgers University, New Brunswick, NJ (1994)
3. Boros, E., Čepek, O., Kogan, A.: Horn minimization by iterative decomposition. Ann. Math. Artif. Intell. **23**, 321–343 (1998)
4. Boros, E., Čepek, O., Kogan, A., Kučera, P.: Exclusive and essential sets of implicates of Boolean functions. Discrete Appl. Math. **158**(2), 81–96 (2010)
5. Boros, E., Čepek, O., Kogan, A., Kučera, P.: A subclass of Horn cnfs optimally compressible in polynomial time. Ann. Math. Artif. Intell. **57**(3–4), 249–291 (2010)
6. Büning, H.K., Letterman, T.: Propositional Logic: Deduction and Algorithms. Cambridge University Press, New York, NY (1999)
7. Čepek, O.: Structural properties and minimization of Horn Boolean functions. Ph.D. thesis, Rutgers University, New Brunswick, NJ (1995)
8. Čepek, O., Kučera, P., Savický, P.: Boolean functions with a simple certificate for CNF complexity. Discrete Applied Mathematics (2011, in print, available online)
9. Dechter, R., Pearl, J.: Structure identification in relational data. Artif. Intell. **58**, 237–270 (1992)

10. Delobel, C., Casey, R.: Decomposition of a data base and the theory of Boolean switching functions. IBM J. Res. Develop. **17**, 374–386 (1973)
11. Genesereth, M., Nilsson, N.: Logical Foundations of Artificial Intelligence. Morgan Kaufmann, Los Altos, CA (1987)
12. Hammer, P., Kogan, A.: Horn functions and their DNFs. IBM J. Res. Develop. **44**, 23–29 (1992)
13. Hammer, P., Kogan, A.: Optimal compression of propositional Horn knowledge bases: complexity and approximation. Artif. Intell. **64**, 131–145 (1993)
14. Hammer, P., Kogan, A.: Knowledge compression—logic minimization for expert systems. In: Computers as our Better Partners. Proceedings of the IISF/ACM Japan International Symposium, pp. 306–312. World Scientific, Singapore (1994)
15. Hammer, P., Kogan, A.: Quasi-acyclic propositional Horn knowledge bases: optimal compression. IEEE Trans. Knowl. Data Eng. **7**(5), 751–762 (1995)
16. Maier, D.: Minimal covers in the relational database model. JACM **27**, 664–674 (1980)
17. Quine, W.: A way to simplify truth functions. Am. Math. Mon. **62**, 627–631 (1955)
18. Russel, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 2nd edn. Pearson Education (2003)
19. Tarjan, R.: Depth first search and linear graph algorithms. SIAM J. Comput. **2**, 146–160 (1972)
20. Umans, C.: The minimum equivalent DNF problem and shortest implicants. J. Comput. Syst. Sci. **63**(4), 597–611 (2001)
21. Umans, C., Villa, T., Sangiovanni-Vincentelli, A.L.: Complexity of two-level logic minimization. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. **25**(7), 1230–1246 (2006)

Discrete Applied Mathematics 160 (2012) 365–382

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Boolean functions with a simple certificate for CNF complexity

Ondřej Čepek [a,b,*], Petr Kučera [a,1], Petr Savický [c]

[a] Department of Theoretical Computer Science and Mathematical Logic, Faculty of Mathematics and Physics, Charles University, Malostranské náměstí 25, 118 00 Praha 1, Czech Republic

[b] Institute of Finance and Administration, Estonska, 500 101 00 Praha 10, Czech Republic

[c] Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodarenskou vezi, 271/2 182 07 Prague 8, Czech Republic

ARTICLE INFO

Article history:
Received 15 January 2010
Received in revised form 13 May 2011
Accepted 31 May 2011
Available online 13 July 2011

Keywords:
Boolean functions
CNF representations

ABSTRACT

In this paper we study relationships between CNF representations of a given Boolean function f and essential sets of implicates of f. It is known that every CNF representation and every essential set must intersect. Therefore the maximum number of pairwise disjoint essential sets of f provides a lower bound on the size of any CNF representation of f. We are interested in functions, for which this lower bound is tight, and call such functions coverable. We prove that for every coverable function there exists a polynomially verifiable certificate (witness) for its minimum CNF size. On the other hand, we show that not all functions are coverable, and construct examples of non-coverable functions. Moreover, we prove that computing the lower bound, i.e. the maximum number of pairwise disjoint essential sets, is *NP*-hard under various restrictions on the function and on its input representation.

1. Introduction

The Boolean minimization (BM) problem can be stated as follows: given a CNF ϕ find a CNF ϕ' representing the same function and such that ϕ' consists of a minimum possible number of clauses. A decision version of the problem is obtained by including a bound in the instance and the question is, whether there is a representation ϕ' of at most the given size. BM has many practical applications. For instance, in artificial intelligence this problem is equivalent to finding the most compact representation of a given knowledge base [11,12]. Such transformation of a knowledge base accomplishes knowledge compression, since the actual knowledge does not change, while the size of the representation can be significantly reduced.

BM is in general a hard problem. Obviously, it contains the satisfiability problem (SAT) as its special case. An unsatisfiable CNF is identically zero, which means that its shortest representation consists only of a constant. In fact, BM was shown to be probably harder than SAT: while SAT is *NP*-complete (i.e. Σ_1^p-complete) [6], the decision version of BM is Σ_2^p-complete [20]. BM remains *NP*-hard even for some classes of Boolean functions for which SAT is solvable in polynomial time. The best known example of such a class are Horn functions (see [2,11,15] for various BM intractability results for the class of Horn functions). The difficulty of BM of course raises a natural question whether for a given input CNF, a nontrivial lower bound can be obtained for the number of clauses in the shortest equivalent CNF. This question was recently addressed in [3] where the concept of essential sets of function f was introduced.

[*] Corresponding author at: Department of Theoretical Computer Science and Mathematical Logic, Faculty of Mathematics and Physics, Charles University, Malostranské náměstí 25, 118 00 Praha 1, Czech Republic. Tel.: +420 221 914 246; fax: +420 221 914 323.
E-mail addresses: ondrej.cepek@mff.cuni.cz, cepek@rutcor.rutgers.edu (O. Čepek), kucerap@ktiml.mff.cuni.cz (P. Kučera), savicky@cs.cas.cz (P. Savický).
[1] Tel.: +420 221 914 138; fax: +420 221 914 323.

0166-218X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2011.05.013

Similarly as in [3], the main object of interest throughout this paper will be the set $\mathcal{I}(f)$ defined as the resolution closure of the set $\mathcal{I}^p(f)$ of all prime implicates of f. A subset $\mathcal{E} \subseteq \mathcal{I}(f)$ is an essential set of f, if $\mathcal{I}(f) \setminus \mathcal{E}$ is closed under resolution. It was shown in [3] that given a Boolean function f, every CNF representation of f must intersect every nonempty essential set. Therefore, the maximum number of pairwise disjoint essential sets constitutes a lower bound on the size of any CNF which represents f.

In this paper we are primarily interested in functions for which the above described lower bound is tight. We shall call such functions coverable. It should be noted that nontrivial subclasses of Boolean functions which consist of coverable functions are already known. These include acyclic and quasi-acyclic Horn functions [3] as well as the class of CQ-Horn functions [5].

After introducing the necessary notation and presenting the basic results from [3] related to essential sets in Section 2, we show in Section 3 that for every coverable function f there exists a polynomially verifiable certificate (witness) for the size of its minimum CNF representation, i.e. a certificate sufficient for a polynomial time verification that no CNF representation of f has fewer clauses than the given minimum one. In Section 4 we study tractable classes of CNFs, and prove that if a tractable class is coverable (i.e. all CNFs in the class represent coverable functions) then the decision version of BM for this class is both in *NP* and *coNP* and derive several consequences of this fact.

Given a CNF which represents a function f, it may be difficult to compute the lower bound (i.e. the maximum number of pairwise disjoint essential sets) simply because the set $\mathcal{I}(f)$ is too large. Therefore we define in Section 5 projections of essential sets on the set $\mathcal{I}^p(f)$ of prime implicates, and show that the lower bound on the size of f can be characterized using these projections only. This allows us to work with smaller sets of implicates and thus prove or disprove the tightness of the lower bound for particular input CNFs more efficiently. Moreover, it is shown in Section 5 that several properties of essential sets carry over to the studied projections. Using the results of Section 5 we construct in Section 6 an example of a function where the lower bound is not tight, and moreover we show that the gap between the lower bound and the size of the minimal CNF can be made arbitrarily large.

In Section 7 we prove that given a CNF which represents a function f, computing the maximum number of pairwise disjoint essential sets of f is *NP*-hard, even if the input is restricted to cubic pure Horn CNFs. Finally, in Section 8, we show that in the unrestricted case, computing the maximum number of pairwise disjoint essential sets is *NP*-hard, when the function is given by its truth table instead of a CNF. On the other hand, given a truth table representation, a relaxation of the lower bound based on linear programming is shown to be obtainable in polynomial time.

2. Basic notation, definitions, and results

In this section we introduce the necessary notation and summarize the basic known results that will be needed later in the text.

2.1. Boolean functions

A *Boolean function* f on n propositional variables $x_1, \ldots, x_n$ is a mapping $\{0, 1\}^n \to \{0, 1\}$. The propositional variables $x_1, \ldots, x_n$ and their negations $\overline{x}_1, \ldots, \overline{x}_n$ are called *literals* (*positive* and *negative literals*, respectively). An elementary disjunction of literals is called a *clause*, if every propositional variable appears in it at most once. A clause C is called an *implicate* of a function f if for every $x \in \{0, 1\}^n$ we have $f(x) \leq C(x)$. An implicate C is called *prime* if dropping any literal from it produces a clause which is not an implicate.

It is a well-known fact that every Boolean function f can be represented by a conjunction of clauses (see e.g. [9]). Such an expression is called a *conjunctive normal form* (or CNF) of the Boolean function f. In the rest of the paper we shall often identify a CNF ϕ with a set of its clauses and we shall use both notions interchangeably. A CNF ϕ representing a function f is called *prime* if each clause of ϕ is a prime implicate of the function f. A CNF ϕ representing a function f is called *irredundant* if dropping any clause from ϕ produces a CNF that does not represent f.

Two clauses C_1 and C_2 are said to be *resolvable* if they contain exactly one complementary pair of literals. That means that we can write $C_1 = \tilde{C}_1 \vee x$ and $C_2 = \tilde{C}_2 \vee \overline{x}$ for some propositional variable x and clauses $\tilde{C}_1$ and $\tilde{C}_2$ which contain no complementary pair of literals. The clauses C_1 and C_2 are called *parent clauses* and the disjunction $R(C_1, C_2) = \tilde{C}_1 \vee \tilde{C}_2$ is called the *resolvent* of the parent clauses C_1 and C_2. Note that the resolvent is a clause (does not contain a propositional variable and its negation). We say that a clause C can be *derived by a series of resolutions* from a CNF ϕ, if there exists a finite sequence $C_1, C_2, \ldots, C_p$ of clauses such that

(1) $C_p = C$, and
(2) for $i = 1, \ldots, p$, either C_i is a clause in ϕ or there exist $j < i$ and $k < i$ such that $C_i = R(C_j, C_k)$.

It is a well-known fact, see for example [4], that a resolvent of two implicates of f is an implicate of f and every prime implicate of f can be derived by a series of resolutions from any CNF representing f. The so-called *Quine's procedure* [4,17,18] takes a CNF φ as an input and outputs the list of all prime implicates of the function represented by φ. Given a set of clauses $\mathcal{C}$, we shall denote by $\mathcal{R}(\mathcal{C})$ the resolution closure of $\mathcal{C}$, i.e. $\mathcal{R}(\mathcal{C})$ is the set of all clauses, which can be derived by a series of resolutions from clauses in $\mathcal{C}$.

For a Boolean function f let us denote by $\mathcal{I}^p(f)$ the set of its prime implicates, and let $\mathcal{I}(f)$ denote the resolution closure of the set of its prime implicates $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}^p(f))$. A clause C is called *negative* if it contains no positive literals. It is called *pure Horn* if it contains exactly one positive literal. To simplify notation, we shall sometimes write a pure Horn clause $C = \bigvee_{x \in S} \overline{x} \vee y$ simply as $C = S \vee y$. Each propositional variable $x \in S$ is called a *subgoal of* C and the propositional variable y is called the *head of* C.

A CNF is called *Horn*, if it contains only negative and pure Horn clauses. A CNF is called *pure Horn*, if it contains only pure Horn clauses. Finally, a Boolean function is called *Horn*, if it has at least one representation by a Horn CNF, and similarly a Boolean function is called *pure Horn*, if it has at least one representation by a pure Horn CNF.

It is known (see [10]) that each prime implicate of a Horn function is either negative or pure Horn, and each prime implicate of a pure Horn function is pure Horn. Thus, in particular, any prime CNF representing a Horn function is Horn, and any prime CNF representing a pure Horn function is pure Horn.

Definition 2.1. A class of CNFs $\mathcal{X}$ will be called *tractable*, if it satisfies the following properties.

- Recognition: Given an arbitrary CNF φ it is possible to decide in polynomial time with respect to the size of φ whether $\varphi \in \mathcal{X}$.
- Satisfiability: Given an arbitrary CNF $\varphi \in \mathcal{X}$ it is possible to decide in polynomial time with respect to the size of φ whether φ is satisfiable.
- Partial assignment: Given an arbitrary CNF $\varphi \in \mathcal{X}$, if ψ is produced from φ by fixing some variables to 0 or 1 and substituting these values into φ, then $\psi \in \mathcal{X}$.
- Prime representations: Given an arbitrary CNF $\varphi \in \mathcal{X}$, if φ represents a function f then all prime CNF representations of f belong to $\mathcal{X}$.

It follows that given a CNF φ from a tractable class, we can decide in polynomial time whether a given clause C is an implicate of φ by substituting the appropriate values (which make C zero) into φ and testing the satisfiability of the resulting formula. This property of tractable classes has two important consequences.

Lemma 2.2. *Let φ be a CNF from a tractable class. Then it is possible to find in polynomial time a prime and irredundant CNF ψ which is equivalent to φ.*

Proof. For every clause C in φ we can delete its literals one by one and test whether the remaining clause is still an implicate of φ. If yes, the literal is deleted permanently, if no, the literal is returned back into the clause. When no literal can be deleted, we have arrived to a prime subclause of C which can replace C in φ. Note that for different orders of literal deletions we may arrive to different prime subclauses of C. After getting a prime CNF we can test for each clause whether it is an implicate of the CNF defined by the remaining clauses. If yes, the clause is redundant and can be deleted, if no, the clause is kept in place. In this way an irredundant CNF is produced. Note again that for different orders of clause deletions we may arrive to different prime and irredundant CNFs (all representing the same function as φ). Since the described procedure amounts to a linear number of tests whether a given clause is an implicate of a given CNF from a tractable class, it follows that the procedure runs in polynomial time with respect to the length of the input CNF. $\square$

Lemma 2.3. *Let φ and ψ be two CNFs from a tractable class. Then it is possible to test in polynomial time whether φ and ψ represent the same Boolean function (are logically equivalent) or not.*

Proof. It suffices to test for each clause C in φ whether it is an implicate of ψ and for each clause C in ψ whether it is an implicate of φ. The two CNFs are logically equivalent if and only if none of these tests fails. $\square$

An example of a tractable class is the class of Horn CNFs, which we use most frequently in the subsequent text.

2.2. Forward chaining procedure

In verifying that a given clause is an implicate of a given pure Horn function, a very useful and simple procedure is the following. Let η be a pure Horn CNF of a pure Horn function h. We shall define a *forward chaining* procedure [13] which associates to any subset Q of the propositional variables of h a set M in the following way. The procedure takes as input the subset Q of propositional variables, initializes the set $M = Q$, and at each step it looks for a pure Horn clause $S \vee y$ in η such that $S \subseteq M$, and $y \notin M$. If such a clause is found, the propositional variable y is included into M, and the search is repeated as many times as possible. The set M output by this procedure will be denoted by $\mathrm{FC}_\eta(Q)$, where η is the input CNF and Q the starting set of variables. It can be shown [11,19] that a clause $C = Q \vee y$ is an implicate of h if and only if $y \in \mathrm{FC}_\eta(Q)$. If η' and η'' are two distinct CNF representations of a given pure Horn function h and if Q is an arbitrary subset of the propositional variables, then $\mathrm{FC}_{\eta'}(Q) = \mathrm{FC}_{\eta''}(Q)$ because η' and η'' have the same set of implicates. Therefore, the set of propositional variables reachable from Q by forward chaining depends only on the underlying function h rather than on a particular CNF representation η. For this reason, we shall also use the expression $\mathrm{FC}_h(Q)$ instead of $\mathrm{FC}_\eta(Q)$ whenever we do not want to refer to a specific CNF.

2.3. Essential sets

In this section we shall define the central notion of this paper, the essential set of clauses, which was introduced in [3].

Definition 2.4 (*[3]*)**.** Given a Boolean function f, a subset $\mathcal{E} \subseteq \mathcal{I}(f)$ is called an *essential set of f* (or simply an *essential set* if f is clear from the context) if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{I}(f)$ the following implication holds:

$$R(C_1, C_2) \in \mathcal{E} \Longrightarrow C_1 \in \mathcal{E} \text{ or } C_2 \in \mathcal{E},$$

i.e. the resolvent belongs to $\mathcal{E}$ only if at least one of the parent clauses is from $\mathcal{E}$.

It is easy to see that a set is essential if and only if its complement is closed under resolution. Hence, we have also the following characterization.

Theorem 2.5 ([3])**.** *A subset $\mathcal{E}$ of $\mathcal{I}(f)$ is an essential set of f iff $\mathcal{I}(f) \setminus \mathcal{E} = \mathcal{R}(\mathcal{I}(f) \setminus \mathcal{E})$.*

Note that the empty set is an essential set of any Boolean function. We shall often use the notion of a minimal (with respect to inclusion) essential set and we shall require that such a set is nonempty. For this reason, we exclude empty set when defining minimal essential set. In particular, we have the following definition.

Definition 2.6. We shall say that an essential set $\mathcal{E}$ is *minimal*, if $\mathcal{E} \neq \emptyset$ and the only essential set which is properly included in $\mathcal{E}$ is an empty set.

Definition 2.7. For a Boolean function f, let $\mathrm{ess}(f)$ be the maximum number of pairwise disjoint nonempty essential sets of implicates and let $\mathrm{cnf}(f)$ be the minimum number of clauses needed to represent f by a CNF.[2]

An important connection between $\mathrm{ess}(f)$ and $\mathrm{cnf}(f)$ was shown in [3].

Theorem 2.8 (*[3]*)**.** *For every Boolean function f, we have* $\mathrm{cnf}(f) \geq \mathrm{ess}(f)$.

In Section 6.1, we demonstrate an example of a Horn Boolean function, for which we have $\mathrm{cnf}(f) > \mathrm{ess}(f)$. On the other hand, many useful functions satisfy $\mathrm{cnf}(f) = \mathrm{ess}(f)$. When this is satisfied, there is a polynomially verifiable certificate for this fact by Theorem 3.8. The main goal of this paper is to investigate the general properties of essential sets and to derive consequences for the properties of the class of functions satisfying $\mathrm{cnf}(f) = \mathrm{ess}(f)$.

Definition 2.9. Let f be a Boolean function. We shall call f *coverable* if $\mathrm{ess}(f) = \mathrm{cnf}(f)$. Let $\mathcal{X}$ be a set (or class) of CNFs. We shall call $\mathcal{X}$ *coverable* if every CNF from $\mathcal{X}$ represents a coverable function.

3. A polynomially verifiable certificate for ess(*f*)

Given a falsepoint t of f, we define

$$\mathcal{E}(t) = \{C \in \mathcal{I}(f) \mid C(t) = 0\}.$$

The assumption that t is a falsepoint of f implies $\mathcal{E}(t) \neq \emptyset$. Moreover, it is easy to verify that $\mathcal{E}(t)$ is an essential set of implicates (for a proof, see Lemma 6.5 in [3]). The set $\mathcal{E}(t)$ will be called a *falsepoint essential set defined by t* (or an *FE set defined by t* for brevity). It is easy to see that not every essential set is an FE set (consider e.g. the entire set $\mathcal{I}(f)$ which is of course an essential set of f—if it contains two clauses containing a pair of complementary literals then no vector t can falsify both such clauses). However, every minimal essential set of implicates is equal to $\mathcal{E}(t)$ for some t.

Theorem 3.1. *Let f be a Boolean function and let $\mathcal{E}$ be a minimal essential set of $\mathcal{I}(f)$, then there is some falsepoint t of f, such that $\mathcal{E} = \mathcal{E}(t)$.*

Proof. Let g be a function represented by clauses in $\mathcal{I}(f) \setminus \mathcal{E}$. Clearly $g \geq f$, because it is represented by implicates of f. Since $\mathcal{E}$ is essential, we have that $\mathcal{R}(\mathcal{I}(f) \setminus \mathcal{E}) = \mathcal{I}(f) \setminus \mathcal{E} \neq \mathcal{I}(f)$ by Theorem 2.5. By Lemma 4.3 in [3], we have that a subset of $\mathcal{I}(f)$ defines the function f if and only if its resolution closure is $\mathcal{I}(f)$. Hence, we have $g \neq f$. Therefore, there is a vector t, such that $g(t) = 1$, while $f(t) = 0$, and hence every implicate $C \in \mathcal{I}(f)$ for which $C(t) = 0$ belongs to $\mathcal{E}$. In other words, $\mathcal{E}(t) \subseteq \mathcal{E}$. Since $\mathcal{E}(t)$ is an essential set of implicates and $\mathcal{E}$ is a minimal essential set of $\mathcal{I}(f)$, we have $\mathcal{E}(t) = \mathcal{E}$. □

Let us use this fact to provide an equivalent characterization of $\mathrm{ess}(f)$.

Corollary 3.2. *Let f be an arbitrary Boolean function, then* $\mathrm{ess}(f)$ *is equal to the maximum number of disjoint FE sets.*

Proof. Let $k = \mathrm{ess}(f)$. The maximum number of disjoint FE sets of clauses is at most k because every FE set (i.e. $\mathcal{E}(t)$ for an arbitrary falsepoint t) is essential. For the opposite inequality, let $\mathcal{E}_1, \ldots, \mathcal{E}_k$ be a family of pairwise disjoint essential sets

[2] The first number is denoted by $\epsilon(f)$ and the second number by $\sigma(f)$ in [3].

of f. For every $i = 1, \ldots, k$ let $\mathcal{E}_i'$ be a minimal essential set, which is a subset of $\mathcal{E}_i$. Using Theorem 3.1 there exists a vector t_i, such that $\mathcal{E}_i' = \mathcal{E}(t_i)$. Hence $\mathcal{E}_1', \ldots, \mathcal{E}_k'$ constitute k disjoint FE sets and so the maximum number of disjoint FE sets of clauses is at least k. $\square$

Let us prove some further properties of FE sets which are used later.

Definition 3.3. Let s, t, r be Boolean vectors of length n. We say that r separates s and t, if for every $i = 1, \ldots, n$, we have $r_i = s_i$ or $r_i = t_i$.

Definition 3.4. Let s, t be Boolean vectors of length n. Then we denote

$$C_{st} = \bigvee_{i \in I(s,t)} \bar{x}_i \vee \bigvee_{i \in O(s,t)} x_i,$$

where sets $I(s, t)$ and $O(s, t)$ are defined as follows

$$I(s, t) = \{i \mid (1 \le i \le n) \wedge s[i] = t[i] = 1\}$$
$$O(s, t) = \{i \mid (1 \le i \le n) \wedge s[i] = t[i] = 0\}.$$

Note that r separates s, t if and only if $C_{st}(r) = 0$.

Lemma 3.5. *Let s and t be two falsepoints of a Boolean function f. Then the following statements are equivalent:*

1. $\mathcal{E}(s) \cap \mathcal{E}(t) \neq \emptyset$.
2. *C_{st} is an implicate of f.*
3. $\mathcal{E}(s) \cap \mathcal{E}(t) \cap \mathcal{I}^p(f) \neq \emptyset$.

Proof. • (1) $\Longrightarrow$ (2): Let us assume that there exists an implicate $C' \in \mathcal{E}(s) \cap \mathcal{E}(t)$. Since $C'(t) = C'(s) = 0$, we have that variables of all positive literals of C' belong to $O(s, t)$ and variables of all negative literals of C' belong to $I(s, t)$. This in turn means that C' is a subclause of C_{st}. Therefore $C' \le C_{st}$ and hence C_{st} is an implicate of f.

- (2) $\Longrightarrow$ (3): Let us assume that C_{st} is an implicate of f. Clearly, C_{st} evaluates to zero on both s and t (it is by its definition the "longest" clause with this property). Since C_{st} is an implicate of f, there exists a prime implicate $C' \in \mathcal{I}^p(f)$ such that $C' \le C_{st}$ (i.e. C' is a subclause of C_{st}). Since C' also evaluates to zero on both s and t, we have $C' \in \mathcal{E}(s) \cap \mathcal{E}(t) \cap \mathcal{I}^p(f)$, which need not be true for C_{st}.
- (3) $\Longrightarrow$ (1): This implication is trivial. $\square$

Note that if the given CNF representation of f is from a tractable class (e.g. if it is a Horn CNF), then for every pair of vectors s and t we can test in polynomial time, whether $\mathcal{E}(t) \cap \mathcal{E}(s) = \emptyset$ or not. This observation easily follows from Lemma 3.5 and the fact that testing whether a given clause is an implicate of a function given by a CNF from a tractable class can be done in polynomial time.

Corollary 3.6. *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be two minimal essential sets of implicates of a Boolean function f, then $\mathcal{E}_1$ and $\mathcal{E}_2$ have a nonempty intersection if and only if there is a prime implicate of f which belongs to both $\mathcal{E}_1$ and $\mathcal{E}_2$.*

Proof. This directly follows from Theorem 3.1 and Lemma 3.5. $\square$

The following formulation explicitly shows a certificate for the disjointness of two FE sets.

Lemma 3.7. *Let s and t be two falsepoints of a Boolean function f. Then $\mathcal{E}(s)$ and $\mathcal{E}(t)$ are disjoint if and only if there exists a truepoint r of f, which separates s and t.*

Proof. Since r separates s, t if and only if $C_{st}(r) = 0$, we obtain that there exists a truepoint r, which separates s, t if and only if C_{st} is not an implicate. Then, the lemma follows by taking negations of parts 1 and 2 in Lemma 3.5. $\square$

Let us now formulate the following decision problem.

Problem ESS($\mathcal{F}, k$).
Input: A CNF $\mathcal{F}$ which represents a Boolean function f and a natural number k.
Question: ess(f) $\ge k$?

Now we shall show that this problem belongs to the class *NP*. In Sections 7 and 8, we shall prove that it is also *NP* hard.

Theorem 3.8. *Problem* ESS($\mathcal{F}, k$) *is in NP.*

Proof. Let a pair $\mathcal{F}, k$ be a positive instance of ESS($\mathcal{F}, k$), i.e. let ess(f) $\ge k$ hold, where f is the Boolean function represented by $\mathcal{F}$. Then by Corollary 3.2 there exist k falsepoints $t_1, \ldots, t_k$ of function f which define pairwise disjoint nonempty FE sets $\mathcal{E}(t_1), \ldots, \mathcal{E}(t_k)$. Let $1 \le i < j \le k$ be arbitrary. By Lemma 3.7 there exists a truepoint r_{ij} of f which separates t_i and t_j. However, now the vectors t_i, $1 \le i \le k$, and r_{ij}, $1 \le i < j \le k$ form a certificate for ess(f) $\ge k$. This certificate has a polynomial size with respect to the input CNF $\mathcal{F}$ because it consists of $O(k^2)$ vectors of length n while $\mathcal{F}$ consists of at least ess(f) $\ge k$ clauses by Theorem 2.8 (and we may assume without loss of generality that each of n variables appears at least once in $\mathcal{F}$). Of course, such a certificate is also polynomially verifiable: it suffices to check that every t_i, $1 \le i \le k$ is a falsepoint of f (by substituting the appropriate binary values into $\mathcal{F}$), and that every r_{ij}, $1 \le i < j \le k$ is a truepoint of f which separates t_i and t_j. $\square$

4. CNF minimization for tractable classes

Let us start this section by formulating CNF minimization as a decision problem.

Problem CNF($\mathcal{F}$, ℓ).
Input: A CNF $\mathcal{F}$ which represents a Boolean function f and a natural number ℓ.
Question: cnf(f) $\leq \ell$?
We shall show that this decision problem is in *NP* when the input CNFs are restricted to some tractable class of CNFs.

Lemma 4.1. *Let $\mathcal{X}$ be a tractable class of CNFs. Then* CNF($\mathcal{F}$, ℓ) *is in NP for $\mathcal{F} \in \mathcal{X}$.*

Proof. Let a pair $\mathcal{F}$, ℓ be a positive instance of CNF($\mathcal{F}$, ℓ), i.e. let cnf(f) $\leq \ell$ hold, where f is the Boolean function represented by $\mathcal{F}$. Then a prime CNF $\mathcal{G}$, which represents f and consists of at most ℓ clauses is a polynomial size certificate for this inequality. Note that we may assume that $\mathcal{G}$ is a prime representation since the existence of a CNF representing f and consisting of at most ℓ clauses clearly implies the existence of a prime CNF with the same property. Moreover, the tractability of $\mathcal{X}$ implies $\mathcal{G} \in \mathcal{X}$. The fact that $\mathcal{G}$ is a polynomially verifiable certificate follows from the fact that both $\mathcal{F}$ and $\mathcal{G}$ belong to the tractable class $\mathcal{X}$, and hence it is possible to test in polynomial time that they both represent the same function f (see Lemma 2.3). □

Theorem 4.2. *Let $\mathcal{X}$ be a class of CNFs which is both tractable and coverable. Then* CNF($\mathcal{F}$, ℓ) *is in NP $\cap$ coNP for $\mathcal{F} \in \mathcal{X}$.*

Proof. The fact that CNF($\mathcal{F}$, ℓ) is in *NP* for $\mathcal{F} \in \mathcal{X}$, i.e. that there exists a polynomially verifiable certificate for a positive answer, follows directly from Lemma 4.1. Let f be the Boolean function represented by $\mathcal{F}$. A certificate for a negative answer is a certificate for the fact that cnf(f) $\geq \ell + 1$ which is the same as ess(f) $\geq \ell + 1$ since cnf(f) = ess(f) due to the fact that f is coverable. However, such a certificate, which is polynomially verifiable, exists due to Theorem 3.8. □

It should be remarked here that the requirement $\mathcal{X} \in P$, i.e. that there exists a polynomial time recognition algorithm for $\mathcal{X}$ (imposed on tractable classes in Definition 2.1), can be weakened to $\mathcal{X} \in$ *NP* $\cap$ *coNP* while both Lemma 4.1 and Theorem 4.2 remain valid.

It can be also pointed out that even a stricter "equality version" of CNF($\mathcal{F}$, ℓ), where the input stays the same but the question is changed to cnf(f) = ℓ?, is still in *NP* $\cap$ *coNP* for $\mathcal{F}$ in a tractable and coverable class. A certificate for a positive answer is a combination of certificates for cnf(f) $\leq \ell$ and ess(f) $\geq \ell$, while a certificate for a negative answer is one of the certificates for cnf(f) $\leq \ell - 1$ or ess(f) $\geq \ell + 1$.

Theorem 4.2 indicates that if for a tractable class $\mathcal{X}$ one can show that $\mathcal{X}$ is coverable, then there is a good chance CNF($\mathcal{F}$, ℓ) is solvable in polynomial time for $\mathcal{F} \in \mathcal{X}$, as most decision problems known to be in *NP* $\cap$ *coNP* are in fact in *P*. This is indeed the case for all three classes known to be simultaneously tractable and coverable which were mentioned in the Introduction (acyclic Horn, quasi-acyclic Horn, and CQ-Horn CNFs). Let us now state a simple corollary.

Corollary 4.3. *Let $\mathcal{X}$ be a tractable class of CNFs for which the minimization problem* CNF($\mathcal{F}$, ℓ) *is NP-hard. Then $\mathcal{X}$ is not coverable unless NP = coNP.*

Proof. Let us proceed by contradiction and assume that $\mathcal{X}$ is coverable. Then by Theorem 4.2 we have that CNF($\mathcal{F}$, ℓ) is in *NP* $\cap$ *coNP* for $\mathcal{F} \in \mathcal{X}$, and by an assumption CNF($\mathcal{F}$, ℓ) is *NP*-hard for $\mathcal{F} \in \mathcal{X}$. However, the fact that an *NP*-complete problem falls into *coNP* implies *NP* = *coNP* (see e.g. [8]). □

There are many classes with *NP*-hard minimization which may play the role of class $\mathcal{X}$ in Corollary 4.3. A good example is the class of Horn CNFs [2]. Therefore, unless *NP* = *coNP*, there must exist a Horn CNF representing function f for which ess(f) < cnf(f). We shall construct such a CNF in Section 6.1 after we introduce further notation and derive results needed to prove the properties of such a CNF. In particular, we shall first concentrate on how to compute ess(f) using only clauses from $\mathcal{I}^p(f)$ instead of looking at the entire $\mathcal{I}(f)$ which may be much larger.

5. Prime essential sets

According to Corollary 3.6, in order to test the disjointness of minimal essential sets, it is sufficient to look at prime clauses. This suggests to consider the following notion.

Definition 5.1. Let f be an arbitrary Boolean function and let $\mathcal{E} \subseteq \mathcal{I}^p(f)$ be a set of prime implicates. We say that $\mathcal{E}$ is a *prime essential set of f* (or simply a *prime essential set* if f is clear from the context), if $\mathcal{E} = \mathcal{E}' \cap \mathcal{I}^p(f)$ for a set of clauses $\mathcal{E}'$ such that $\mathcal{E}'$ is an essential set of f. We shall say that a prime essential set $\mathcal{E}$ of f is *minimal*, if $\mathcal{E} \neq \emptyset$ and the only prime essential set of f which is properly included in $\mathcal{E}$ is the empty set.

Note that every nonempty essential set $\mathcal{E}'$ contains at least one prime implicate (otherwise the complement $\mathcal{I}(f) \setminus \mathcal{E}'$ contains all prime implicates implying $\mathcal{R}(\mathcal{I}(f) \setminus \mathcal{E}') = \mathcal{I}(f)$ and thus contradicting Theorem 2.5), so prime essential set $\mathcal{E} = \mathcal{E}' \cap \mathcal{I}^p(f)$ is nonempty whenever $\mathcal{E}'$ is nonempty.

In order to characterize prime essential sets in a way similar to the characterization of essential sets in Theorem 2.5, we introduce the following notation for the resolution closure restricted to prime clauses. Moreover, to make the presentation in subsequent sections simpler, we extend this notation also to FE sets.

Definition 5.2. Let f be an arbitrary Boolean function. For every set $\mathcal{C}$ of prime implicates of f, let $\mathcal{R}^p(\mathcal{C}) = \mathcal{R}(\mathcal{C}) \cap \mathcal{I}^p(f)$. For every FE set $\mathcal{E}(t)$ let $\mathcal{E}^p(t) = \mathcal{E}(t) \cap \mathcal{I}^p(f)$.

Theorem 5.3. *A subset $\mathcal{E}$ of $\mathcal{I}^p(f)$ is a prime essential set of f if and only if $\mathcal{I}^p(f) \setminus \mathcal{E} = \mathcal{R}^p(\mathcal{I}^p(f) \setminus \mathcal{E})$.*

Proof. First, assume that $\mathcal{E}$ is a prime essential set. Then, there is an essential set $\mathcal{E}'$ such that $\mathcal{E} = \mathcal{E}' \cap \mathcal{I}^p(f)$ and $\mathcal{E}'$ satisfies $\mathcal{I}(f) \setminus \mathcal{E}' = \mathcal{R}(\mathcal{I}(f) \setminus \mathcal{E}')$ by Theorem 2.5. Since $\mathcal{I}^p(f) \setminus \mathcal{E} \subseteq \mathcal{I}(f) \setminus \mathcal{E}'$, we have $\mathcal{R}^p(\mathcal{I}^p(f) \setminus \mathcal{E}) \subseteq \mathcal{R}^p(\mathcal{I}(f) \setminus \mathcal{E}') = (\mathcal{I}(f) \setminus \mathcal{E}') \cap \mathcal{I}^p(f) = \mathcal{I}^p(f) \setminus \mathcal{E}$. On the other hand, we have $\mathcal{I}^p(f) \setminus \mathcal{E} \subseteq \mathcal{R}^p(\mathcal{I}^p(f) \setminus \mathcal{E})$. Altogether, $\mathcal{R}^p(\mathcal{I}^p(f) \setminus \mathcal{E}) = \mathcal{I}^p(f) \setminus \mathcal{E}$.

For the opposite direction, assume $\mathcal{I}^p(f) \setminus \mathcal{E} = \mathcal{R}^p(\mathcal{I}^p(f) \setminus \mathcal{E})$ and define $\mathcal{E}' = \mathcal{I}(f) \setminus \mathcal{R}(\mathcal{I}^p(f) \setminus \mathcal{E})$. We have $\mathcal{I}(f) \setminus \mathcal{E}' = \mathcal{R}(\mathcal{I}^p(f) \setminus \mathcal{E})$ and, hence, $\mathcal{R}(\mathcal{I}(f) \setminus \mathcal{E}') = \mathcal{R}(\mathcal{R}(\mathcal{I}^p(f) \setminus \mathcal{E})) = \mathcal{R}(\mathcal{I}^p(f) \setminus \mathcal{E}) = \mathcal{I}(f) \setminus \mathcal{E}'$. Consequently, by Theorem 2.5, $\mathcal{E}'$ is an essential set. Since $(\mathcal{I}(f) \setminus \mathcal{E}') \cap \mathcal{I}^p(f) = \mathcal{R}^p(\mathcal{I}^p(f) \setminus \mathcal{E}) = \mathcal{I}^p(f) \setminus \mathcal{E}$, we also have $\mathcal{E}' \cap \mathcal{I}^p(f) = \mathcal{E}$ and $\mathcal{E}$ is a prime essential set. □

Theorem 5.4. *Let f be an arbitrary Boolean function. Then* ess(f) *is equal to the maximum number of pairwise disjoint prime essential sets of f.*

Proof. Let k be the maximum number of pairwise disjoint prime essential sets, and let $\mathcal{E}_1, \dots, \mathcal{E}_{\mathrm{ess}(f)}$ be disjoint essential sets. Since $\mathcal{E}_i \cap \mathcal{I}^p$ are disjoint prime essential sets, we have $k \geq \mathrm{ess}(f)$.

Let $\mathcal{E}_i$ for $i = 1, \dots, k$ be disjoint prime essential sets. Consider minimal essential sets $\mathcal{E}'_i$ such that $\mathcal{E}_i \supseteq \mathcal{E}'_i \cap \mathcal{I}^p$. If $\mathcal{E}'_i$ and $\mathcal{E}'_j$ for $i \neq j$ are not disjoint, then by Corollary 3.6 their intersection contains a prime implicate. This is a contradiction with the assumption that $\mathcal{E}_i$ and $\mathcal{E}_j$ are disjoint. Hence, $\mathcal{E}'_1, \dots, \mathcal{E}'_k$ are disjoint and we have $k \leq \mathrm{ess}(f)$. □

The following lemma gives a connection between minimal essential sets and minimal prime essential sets.

Lemma 5.5. *Let f be an arbitrary Boolean function, and let $\mathcal{E}_p$ be a minimal prime essential set of implicates of f, then there exist a minimal essential set $\mathcal{E}$ of implicates of f, such that $\mathcal{E}_p = \mathcal{E} \cap \mathcal{I}^p(f)$.*

Proof. By definition, there is an essential set $\mathcal{E}$ such that $\mathcal{E}_p = \mathcal{E} \cap \mathcal{I}^p$. Consider any minimal essential subset $\mathcal{E}'$ of $\mathcal{E}$. The intersection $\mathcal{E}' \cap \mathcal{I}^p$ is a nonempty subset of $\mathcal{E}_p$, which is a prime essential set. Since $\mathcal{E}_p$ is minimal, we have $\mathcal{E}' \cap \mathcal{I}^p = \mathcal{E}_p$. □

Note that the reverse direction of Lemma 5.5 does not hold in general, i.e. given a minimal essential set $\mathcal{E}$ of implicates of a Boolean function f, we cannot conclude that $\mathcal{E} \cap \mathcal{I}^p(f)$ is a minimal prime essential set. Consider the function f defined by the following set of clauses

$$\mathcal{F} = \{(a \vee b \vee \overline{c}), (a \vee b \vee \overline{d}), (c \vee \overline{d}), (c \vee \overline{e}), (b \vee \overline{c} \vee d), (a \vee b \vee \overline{e}), (\overline{c} \vee d \vee e), (b \vee d \vee \overline{e})\}.$$

The following set of additional clauses can be derived from $\mathcal{F}$ by resolution

$$\mathcal{G} = \{(a \vee b \vee \overline{c} \vee e), (a \vee b \vee \overline{c} \vee d), (a \vee b \vee d \vee \overline{e}),$$
$$(a \vee b \vee \overline{d} \vee e), (a \vee b \vee c \vee \overline{d}), (a \vee b \vee c \vee \overline{e}), (b \vee c \vee \overline{e})\}.$$

Notice that every clause in $\mathcal{G}$ contains some subclause from $\mathcal{F}$ which means that $\mathcal{I}^p(f) = \mathcal{F}$, and $\mathcal{I}(f) = \mathcal{F} \cup \mathcal{G}$. To verify this fact it suffices to check that for every pair of resolvable clauses from $\mathcal{F} \cup \mathcal{G}$ the resolvent already belongs to $\mathcal{F} \cup \mathcal{G}$. Moreover, if we denote $t_1 = (00111)$ and $t_2 = (00110)$, it can be checked that the sets of clauses

$$\mathcal{E}(t_1) = \{(a \vee b \vee \overline{c}), (a \vee b \vee \overline{d}), (a \vee b \vee \overline{e})\}$$
$$\mathcal{E}(t_2) = \{(a \vee b \vee \overline{c}), (a \vee b \vee \overline{d}), (a \vee b \vee \overline{c} \vee e), (a \vee b \vee \overline{d} \vee e)\}$$

are minimal essential sets of f. However, the sets

$$\mathcal{E}^p(t_1) = \mathcal{E}(t_1) \cap \mathcal{I}^p(f) = \mathcal{E}(t_1)$$
$$\mathcal{E}^p(t_2) = \mathcal{E}(t_2) \cap \mathcal{I}^p(f) = (a \vee b \vee \overline{c})(a \vee b \vee \overline{d}),$$

satisfy $\mathcal{E}^p(t_2) \subsetneq \mathcal{E}^p(t_1)$, and therefore $\mathcal{E}^p(t_1)$ is not a minimal prime essential set.

Corollary 5.6. *Let f be an arbitrary Boolean function and let $\mathcal{E}_p$ be a minimal prime essential set of implicates of f, then there is a falsepoint t of f, such that $\mathcal{E}_p = \mathcal{E}^p(t)$.*

Proof. According to Lemma 5.5 there is a minimal essential set $\mathcal{E}$ such that $\mathcal{E}_p = \mathcal{E} \cap \mathcal{I}^p(f)$. According to Theorem 3.1 there is a falsepoint t of f for which $\mathcal{E} = \mathcal{E}(t)$ and thus $\mathcal{E}_p = \mathcal{E}(t) \cap \mathcal{I}^p(f) = \mathcal{E}^p(t)$. □

The following theorem appears in [3] as two parts, one implication as Corollary 6.14 and the other as Theorem 6.15.

Theorem 5.7 ([3])**.** *Let f be an arbitrary Boolean function and let $\mathcal{E} \subseteq \mathcal{I}(f)$ be an arbitrary set of clauses. Then $\mathcal{E}$ is a minimal essential set iff $\mathcal{E}$ is a minimal (with respect to inclusion) subset of $\mathcal{I}(f)$ such that $\mathcal{E} \cap \mathcal{C} \neq \emptyset$ for every $\mathcal{C} \subseteq \mathcal{I}(f)$ which represents f.*

We can now state a similar result for prime essential sets.

Theorem 5.8. *Let f be an arbitrary Boolean function and let $\mathcal{E}_p \subseteq \mathcal{I}^p(f)$ be an arbitrary set of prime implicates of f. Then $\mathcal{E}_p$ is a minimal prime essential set iff $\mathcal{E}_p$ is a minimal (with respect to inclusion) subset of $\mathcal{I}(f)$ such that $\mathcal{E}_p \cap \mathcal{C} \neq \emptyset$ for every $\mathcal{C} \subseteq \mathcal{I}^p(f)$ which represents f.*

Proof. First, let us prove both directions of the equivalence without proving the minimality of the corresponding set in the conclusion.

Let $\mathcal{E}_p$ be a minimal prime essential set and let $\mathcal{C} \subseteq \mathcal{I}^p(f)$ be an arbitrary prime representation of f. By Lemma 5.5 there exists a minimal essential set $\mathcal{E}$ such that $\mathcal{E}_p = \mathcal{E} \cap \mathcal{I}^p(f)$. Theorem 5.7 now implies that $\mathcal{E} \cap \mathcal{C} \neq \emptyset$. This fact together with the assumption $\mathcal{C} \subseteq \mathcal{I}^p(f)$ gives us $\mathcal{E}_p \cap \mathcal{C} \neq \emptyset$. Since $\mathcal{C}$ was an arbitrary prime representation of f we get that $\mathcal{E}_p \cap \mathcal{C} \neq \emptyset$ for every $\mathcal{C} \subseteq \mathcal{I}^p(f)$ which represents f.

Now let us assume that $\mathcal{E}_p \cap \mathcal{C} \neq \emptyset$ for every $\mathcal{C} \subseteq \mathcal{I}^p(f)$ which represents f and that $\mathcal{E}_p$ is a minimal subset of $\mathcal{I}(f)$ with this property. It follows that $\mathcal{E}_p \subseteq \mathcal{I}^p(f)$. Let us show that $\mathcal{E}_p$ is a prime essential set. Let $\mathcal{E}' \subseteq \mathcal{I}(f) \setminus \mathcal{I}^p(f)$ be a minimal (with respect to inclusion) set of nonprime implicates such that $(\mathcal{E}' \cup \mathcal{E}_p) \cap \mathcal{C} \neq \emptyset$ for every $\mathcal{C} \subseteq \mathcal{I}(f)$ which represents f. Since $\mathcal{E}_p$ already intersects every prime representation of f, adding nonprime implicates is sufficient. By construction, $\mathcal{E} = \mathcal{E}' \cup \mathcal{E}_p$ is a minimal subset of $\mathcal{I}(f)$ such that $\mathcal{E} \cap \mathcal{C} \neq \emptyset$ for every $\mathcal{C} \subseteq \mathcal{I}(f)$ which represents f and thus by Theorem 5.7 we obtain that $\mathcal{E}$ is a minimal essential set of f. Moreover, $\mathcal{E}_p = \mathcal{E} \cap \mathcal{I}^p(f)$ holds and hence $\mathcal{E}_p$ is a prime essential set.

It remains to show that in both directions, we get, in fact, minimal sets. Let $\mathcal{E}_p$ be a minimal prime essential set. By the first paragraph of the proof we know that this set intersects any prime representation of f. If there is a proper subset of $\mathcal{E}_p$, which also intersects every prime representation, then by the second paragraph, this subset is a prime essential set. This is not possible, since $\mathcal{E}_p$ was a minimal prime essential set.

Now, let $\mathcal{E}_p$ be an inclusion minimal set, which intersects every prime representation. By the second paragraph of the proof, we know that it is a prime essential set. If there is a proper subset of $\mathcal{E}_p$, which is also a prime essential set, then, by the first paragraph of the proof, it also intersects every prime representation. This is not possible, since $\mathcal{E}_p$ is an inclusion minimal set with this property. $\square$

Now let us recall Theorem 6.6 from [3]. The following theorem is a minor strengthening of that theorem, which uses the fact that a set intersects every nonempty essential set if and only if it intersects every minimal essential set.

Theorem 5.9 ([3]). *Let f be an arbitrary Boolean function. A set $\mathcal{C} \subseteq \mathcal{I}(f)$ is a representation of f iff $\mathcal{C}$ intersects every minimal essential set of f.*

This statement gives a direct corollary for prime essential sets.

Corollary 5.10. *Let f be an arbitrary Boolean function. A set $\mathcal{C} \subseteq \mathcal{I}^p(f)$ is a representation of f iff $\mathcal{C}$ intersects every minimal prime essential set of f.*

In the following section we shall use Corollary 5.10 in the following way: for a given function f we first list all minimal prime essential sets of f, and then use this list to compute how many clauses are needed to intersect every set in the list, i.e. to compute cnf(f).

6. Examples of functions with cnf(f) > ess(f)

In the end of Section 4 we have noticed that unless $NP = coNP$, there must exist a Horn CNF representing function f for which ess(f) < cnf(f). We shall start this section by constructing such a CNF.

6.1. Cubic pure Horn example on 4 variables

Let us consider pure Horn clauses $C_1 = (x_1 \vee \bar{x}_2 \vee \bar{x}_3)$, $C_2 = (\bar{x}_1 \vee x_2 \vee \bar{x}_3)$, $C_3 = (\bar{x}_1 \vee \bar{x}_2 \vee x_3)$, $Q_1 = (\bar{y} \vee x_1)$, $Q_2 = (\bar{y} \vee x_2)$, $Q_3 = (\bar{y} \vee x_3)$ and function f defined by CNF $\mathcal{F} = C_1 \wedge C_2 \wedge C_3 \wedge Q_1 \wedge Q_2 \wedge Q_3$. Notice that each pair among the three cubic clauses C_1, C_2, C_3 has two complementary pairs of literals and hence no such pair of clauses is resolvable. Moreover, no pair among the three quadratic clauses Q_1, Q_2, Q_3 has a complementary pair of literals and thus again no such pair of clauses is resolvable. In fact, there are only six resolvable pairs in the set $\mathcal{S} = \{C_1, C_2, C_3, Q_1, Q_2, Q_3\}$ (all of them "mixed pairs" of one cubic and one quadratic clause), namely (C_1, Q_2), (C_1, Q_3), (C_2, Q_1), (C_2, Q_3), (C_3, Q_1), (C_3, Q_2). It is easy to check that each of the six resolvents is absorbed by some other clause in $\mathcal{S}$ (e.g. the resolvent $x_1 \vee \bar{y} \vee \bar{x}_3$ of the pair (C_1, Q_2) is absorbed by Q_1). Thus, using Quine's resolution procedure to obtain the set of all prime implicates (canonical CNF) of a function defined by CNF $\mathcal{F}$ (this procedure is described, e.g. in [4]), it follows that $\mathcal{I}^p(f) = \mathcal{S}$.

Consider the vectors $t_1 = (0, 1, 1, 0)$, $t_2 = (1, 0, 1, 0)$, $t_3 = (1, 1, 0, 0)$, $t_4 = (0, 0, 1, 1)$, $t_5 = (1, 0, 0, 1)$, $t_6 = (0, 1, 0, 1)$ as truth value assignments of the variables x_1, x_2, x_3, y. These vectors define the following prime essential sets of clauses.

$$\mathcal{E}(t_1) = \{C_1\}$$
$$\mathcal{E}(t_2) = \{C_2\}$$
$$\mathcal{E}(t_3) = \{C_3\}$$
$$\mathcal{E}(t_4) = \{Q_1, Q_2\}$$
$$\mathcal{E}(t_5) = \{Q_2, Q_3\}$$
$$\mathcal{E}(t_6) = \{Q_1, Q_3\}.$$

It is obvious that $\mathcal{E}(t_1), \mathcal{E}(t_2), \mathcal{E}(t_3)$ are minimal prime essential sets as they contain one clause each. To see that also $\mathcal{E}(t_4), \mathcal{E}(t_5), \mathcal{E}(t_6)$ are minimal prime essential sets it suffices to check that the sets $\mathcal{I}^p(f)\setminus\{Q_1\}$, $\mathcal{I}^p(f)\setminus\{Q_2\}$, and $\mathcal{I}^p(f)\setminus\{Q_3\}$, are not closed under $\mathcal{R}^p$, which by Theorem 5.3 implies that none of the sets $\{Q_1\}$, $\{Q_2\}$, and $\{Q_3\}$ is a prime essential set. Moreover, this observation immediately implies that every nonempty prime essential set must contain either one of the cubic clauses C_1, C_2, C_3 or two of the quadratic clauses Q_1, Q_2, Q_3. In other words every nonempty prime essential set must contain one of $\mathcal{E}(t_1), \ldots, \mathcal{E}(t_6)$, which in turn implies that $\mathcal{E}(t_1), \ldots, \mathcal{E}(t_6)$ is a complete list of minimal prime essential sets of f.

Now, using Corollary 5.10 we obtain that $\text{cnf}(f) = 5$. Indeed, all three cubic clauses must be present in $\mathcal{C}$ to intersect $\mathcal{E}(t_1), \mathcal{E}(t_2), \mathcal{E}(t_3)$ and a single quadratic clause is not sufficient to intersect all of $\mathcal{E}(t_4), \mathcal{E}(t_5), \mathcal{E}(t_6)$. Thus we need a minimum of two quadratic clauses which yields the only three minimum cardinality prime representations of f as follows:

$$\varphi_1 = C_1 \wedge C_2 \wedge C_3 \wedge Q_1 \wedge Q_2$$
$$\varphi_2 = C_1 \wedge C_2 \wedge C_3 \wedge Q_2 \wedge Q_3$$
$$\varphi_3 = C_1 \wedge C_2 \wedge C_3 \wedge Q_1 \wedge Q_3.$$

It can now also be easily checked that there are at most 4 pairwise disjoint minimal prime essential sets of implicates of f ($\mathcal{E}(t_1), \mathcal{E}(t_2), \mathcal{E}(t_3)$ together with one of $\mathcal{E}(t_4), \mathcal{E}(t_5), \mathcal{E}(t_6)$) which implies that there are at most 4 pairwise disjoint nonempty prime essential sets of implicates of f and using Theorem 5.4 we get that $\text{ess}(f) = 4$.

The just constructed example has a gap $\text{cnf}(f) - \text{ess}(f) = 5 - 4 = 1$. In the following section we shall show that the gap $\text{cnf}(f) - \text{ess}(f)$ can be made arbitrarily large.

6.2. More general example

Let x_A be a set of n_1 variables and y_B a set of n_2 variables, where A, B are disjoint sets of indices and $n_1 = 2k - 1$ for some integer k. Let us define a function f_{n_1,n_2} of $n = n_1 + n_2$ variables by

$$f_{n_1,n_2}(x_A, y_B) = \left(\left(\bigvee_{i\in B} y_i \vee \sum_{i\in A} x_i \geq k\right) \Rightarrow \bigwedge_{i\in A} x_i\right)$$

or, equivalently, by the following CNF

$$f_{n_1,n_2}(x_A, y_B) = \left(\bigwedge_{\substack{i\in A\\ j\in B}} (\overline{y_j} \vee x_i)\right)\left(\bigwedge_{\substack{A'\subseteq A,\ |A'|=k\\ j\in A\setminus A'}} \left(\bigvee_{i\in A'} \overline{x_i} \vee x_j\right)\right).$$

Lemma 6.1. *The clauses of the above representation of f_{n_1,n_2} form exactly the list of its prime implicates. In other words, the list of prime implicates of f_{n_1,n_2} is formed by the following two types of clauses.*

1. *For every $i \in A$ and $j \in B$, the clause $\overline{y_j} \vee x_i$.*
2. *For every $A' \subseteq A$ satisfying $|A'| = k$ and every $j \in A \setminus A'$, the clause $\bigvee_{i\in A'} \overline{x_i} \vee x_j$.*

Proof. Resolution may be applied either to two clauses of type 2 or to a clause of type 1 and a clause of type 2. One may verify by case inspection that in the former case, the result is either a clause of type 2 or a superset of some of these clauses. In the latter case, the result is a superset of a clause of type 1. Consequently, the set of clauses from the lemma is stable under Quine's procedure [4] and hence, is a list of all prime implicates of the function defined by the conjunction of its elements. □

Theorem 6.2. *The following two types of sets of clauses represent exactly all minimal prime essential sets for f_{n_1,n_2}.*

1. *For every $j \in B$ and every $A' \subseteq A$, such that $|A'| = k$, let $P(j, A')$ be the set of prime implicates*

$$\{\overline{y_j} \vee x_i \mid i \in A'\}.$$

2. *For every $A' \subseteq A$ satisfying $|A'| \in [k, n_1 - 1]$ let $Q(A')$ be the set of prime implicates*

$$\left\{\bigvee_{i\in A''} \overline{x_i} \vee x_j \mid A'' \subseteq A', |A''| = k,\ j \in A \setminus A'\right\}.$$

Proof. Let us describe the falsepoints t of f_{n_1,n_2}, for which $\mathcal{E}^p(t)$ is minimal. It follows from the definition of the function f_{n_1,n_2} that a vector $t \in \{0,1\}^{n_1+n_2}$ is a falsepoint iff t satisfies at least one of the following conditions:

(1) There are $i \in A$ and $j \in B$ such that $t[x_i] = 0$ and $t[y_j] = 1$, or
(2) $\sum_{i\in A} t[x_i] \in [k, n_1 - 1]$.

If a falsepoint t satisfies both (1) and (2), then $\mathcal{E}^p(t)$ is not minimal, since if t' differs from t by setting $t[y_j] = 0$ for every $j \in B$, then t' is again a falsepoint and $\mathcal{E}^p(t') \subsetneq \mathcal{E}^p(t)$. If t is a falsepoint satisfying (1) and not (2), then according to Lemma 6.1

$$\mathcal{E}^p(t) \subseteq \{\overline{y_j} \vee x_i \mid i \in A \text{ and } j \in B\}.$$

If t' is a falsepoint satisfying (2) and not (1), then according to Lemma 6.1

$$\mathcal{E}^p(t') \subseteq \left\{ \bigvee_{i\in A'} \overline{x_i} \vee x_j \mid A' \subseteq A, |A'| = k \text{ and } j \in A \setminus A' \right\}.$$

It follows that for any such pair of falsepoints t and t' we have $\mathcal{E}^p(t) \cap \mathcal{E}^p(t') = \emptyset$, hence, we may consider the candidates for falsepoints t with minimal $\mathcal{E}^p(t)$ in these two groups separately.

Falsepoints satisfying (1) *and not* (2)

Let t be an arbitrary falsepoint satisfying (1) and not (2) such that $\mathcal{E}^p(t)$ is a minimal prime essential set. It follows that

$$\sum_{i\in A} t[x_i] < k.$$

If $\sum_{i\in A} t[x_i] < k - 1$, then we can produce a falsepoint t' from t by setting $t'[x_i] = 1$ for some $i \in A$ for which $t[x_i] = 0$. Falsepoint t' still satisfies (1) and not (2) and $\mathcal{E}^p(t') \subsetneq \mathcal{E}^p(t)$, because there are fewer unsatisfied clauses of form $\overline{y_j} \vee x_i$ on t'. Since we assume that $\mathcal{E}^p(t)$ is a minimal prime essential set, we get

$$\sum_{i\in A} t[x_i] = k - 1.$$

Similarly, if there are at least two indices $j_1, j_2 \in B$ such that $t[y_{j_1}] = t[y_{j_2}] = 1$, then we can produce falsepoint t' from t by setting $t'[y_{j_2}] = 0$. Falsepoint t' again satisfies (1) and not (2) and $\mathcal{E}^p(t') \subsetneq \mathcal{E}^p(t)$ because there are fewer unsatisfied clauses of form $\overline{y_j} \vee x_i$ on t'. Since we assume that $\mathcal{E}^p(t)$ is a minimal prime essential set this is not possible, and thus there is exactly one $j \in B$ such that $t[y_j] = 1$. Together we have that a falsepoint t which satisfies (1) and not (2) defines a minimal prime essential sets only if the following two conditions are satisfied:

(a) There is exactly one $j \in B$ such that $t[y_j] = 1$ and
(b) $\sum_{i\in A} t[x_i] = k - 1$.

Now we shall show that these two conditions are also sufficient for t to define a minimal prime essential set $\mathcal{E}^p(t)$.

Let t and t' be two different falsepoints satisfying (a) and (b), and let us assume that $j_1 \in B$ is the only index such that $t[y_{j_1}] = 1$ and that $j_2 \in B$ is the only index such that $t'[y_{j_2}] = 1$. We shall show that $\mathcal{E}^p(t)$ is incomparable with $\mathcal{E}^p(t')$. If $j_1 \neq j_2$ then clearly $\mathcal{E}^p(t) \cap \mathcal{E}^p(t') = \emptyset$ so let us suppose that $j_1 = j_2$. Since $t \neq t'$ but $\sum_{i\in A} t[x_i] = \sum_{i\in A} t'[x_i] = k - 1$, we have that there are $i_1, i_2 \in A$ for which $t[x_{i_1}] = 0$, $t[x_{i_2}] = 1$, $t'[x_{i_1}] = 1$, and $t'[x_{i_2}] = 0$. Clearly $\overline{y_{j_1}} \vee x_{i_1} \in \mathcal{E}^p(t) \setminus \mathcal{E}^p(t')$ and $\overline{y_{j_1}} \vee x_{i_2} \in \mathcal{E}^p(t') \setminus \mathcal{E}^p(t)$, and therefore $\mathcal{E}^p(t)$ and $\mathcal{E}^p(t')$ are incomparable. It follows that every falsepoint t satisfying both (a) and (b) defines a minimal prime essential set $\mathcal{E}^p(t)$.

Let t be a falsepoint satisfying (a) and (b), let $j \in B$ be the only index for which $t[y_j] = 1$, and let us denote $A' = \{i \in A \mid t[x_i] = 0\}$. Since $n_1 = 2k - 1$ we have that $|A'| = k$ and thus

$$\mathcal{E}^p(t) = \{\overline{y_j} \vee x_i \mid i \in A'\} = P(j, A').$$

On the other hand, if $j \in B$ and $A' \subseteq A$ with $|A'| = k$ one can easily construct a falsepoint t which satisfies (a) and (b) and for which $\mathcal{E}^p(t) = P(j, A')$ as follows:

$$\begin{array}{ll} t[y_{j'}] = 0 & j' \in B \setminus \{j\} \\ t[y_j] = 1 & \\ t[x_i] = 0 & i \in A' \\ t[x_i] = 1 & i \in A \setminus A'. \end{array}$$

Therefore, the sets $P(j, A')$ are in one-to-one correspondence with falsepoints satisfying (a) and (b). Note also that since $k > n_1/2$ we have that $P(j, A') \cap P(j, A'') \neq \emptyset$ for every $j \in B$ and $A', A'' \subseteq A$ where $|A'| = |A''| = k$.

Falsepoints satisfying (2) *and not* (1)

If t is a falsepoint satisfying (2) and not (1) then $t[y_j] = 0$ for every $j \in B$. Let us show that for any two such falsepoints t_1 and t_2 the sets $\mathcal{E}^p(t_1)$ and $\mathcal{E}^p(t_2)$ are incomparable. Let $A'_i = \{j \in A \mid t_i[x_j] = 1\}$, $i = 1, 2$, we have $|A'_i| \in [k, n_1 - 1]$ for $i = 1, 2$. We shall distinguish two cases, whether A'_1 and A'_2 are comparable, or not.

If A'_1 and A'_2 are incomparable, choose some $a_1 \in A'_1 \setminus A'_2$ and $a_2 \in A'_2 \setminus A'_1$. Further, let A''_1 be an arbitrary subset of A'_1 of size k and similarly, A''_2 a subset of A'_2 of size k. Then the clause

$$\bigvee_{i \in A''_1} \overline{x_i} \vee x_{a_2} \tag{1}$$

evaluates to 0 on t_1 and to 1 on t_2. Similarly, the clause

$$\bigvee_{i \in A''_2} \overline{x_i} \vee x_{a_1} \tag{2}$$

evaluates to 1 on t_1 and to 0 on t_2. Consequently, $\mathcal{E}^p(t_1)$ and $\mathcal{E}^p(t_2)$ are incomparable.

If A'_1 and A'_2 are comparable, assume w.l.o.g. $A'_1 \subsetneq A'_2$ and let $a_2 \in A'_2 \setminus A'_1$. Then, there is a clause constructed similarly to (1), which evaluates to 0 on t_1 and to 1 on t_2. On the other hand, if A''_2 is a k element subset of A'_2, which is not a subset of A'_1, and a_1 is an arbitrary index not contained in A'_2, then the clause of the form (2) evaluates to 1 on t_1 and to 0 on t_2. Consequently, $\mathcal{E}^p(t_1)$ and $\mathcal{E}^p(t_2)$ are incomparable.

It follows that given a falsepoint t satisfying (2) and not (1), $\mathcal{E}^p(t)$ is a minimal prime essential set, if we denote $A' = \{i \in A \mid t[x_i] = 1\}$, then $|A'| \in [k, n_1 - 1]$ and

$$\mathcal{E}^p(t) = \left\{ \bigvee_{i \in A''} \overline{x_i} \vee x_j \mid A'' \subseteq A', |A''| = k,\ j \in A \setminus A' \right\}$$

and therefore $\mathcal{E}^p(t) = Q(A')$. On the other hand, given $A' \subseteq A$, $|A'| \in [k, n_1 - 1]$ we can easily define a falsepoint t satisfying (2) and not (1) for which $\mathcal{E}^p(t) = Q(A')$ by setting $t[x_i] = 1$ iff $i \in A'$ and $t[y_j] = 0$ for every $j \in B$. Thus the sets $Q(A')$ are in one-to-one correspondence with falsepoints satisfying (2) and not (1).

Conclusion

By considering the two cases above, we obtained that the list of prime essential sets presented in the theorem is the list of all minimal prime essential sets of f_{n_1,n_2}. □

Lemma 6.3. *Let n, k be integers such that $n \geq 3$ and $1 \leq k \leq n - 1$. Let A be a set of size n. Let $G_{n,k}$ be the undirected graph of subsets of A of size k, where two sets A' and A'' form an edge if and only if their symmetric difference has size 2. Then $G_{n,k}$ contains a Hamiltonian cycle.*

Proof. If $n \geq 3$ and $k = 1$ or $k = n - 1$, then $G_{n,k}$ is a complete graph, so it contains Hamiltonian cycle. In particular, this proves the lemma for $n = 3$. Let us continue by induction on n.

Assume $n > 3$. It is sufficient to prove the statement for k satisfying $2 \leq k \leq n - 2$. The vertices of $G_{n,k}$ are subsets of $\{1, \ldots, n\}$. Let G_1 be the subgraph of $G_{n,k}$ induced by the vertices containing 1, and let G_2 be the subgraph of $G_{n,k}$ induced by the remaining vertices. Graph G_1 is isomorphic to $G_{n-1,k-1}$ and G_2 is isomorphic to $G_{n-1,k}$. Hence, by induction hypothesis, both G_1 and G_2 contain a Hamiltonian cycle. Fix a Hamiltonian cycle in G_2 and choose an edge (A', A'') contained in it. Let u be an element in the intersection of A' and A''. Such an element exists, since the intersection has size $k - 1$ and $k \geq 2$. Let $B' = A' \setminus \{u\} \cup \{1\}$ and $B'' = A'' \setminus \{u\} \cup \{1\}$. Sets B' and B'' are vertices of G_1 connected by an edge. Since each edge in G_1 may be mapped to any other edge in G_1 by an isomorphism of G_1, there is a Hamiltonian cycle in G_1 containing the edge (B', B''). By removing the edges (A', A'') and (B', B'') from the two Hamiltonian cycles and by connecting them using edges (A', B') and (A'', B''), we obtain a Hamiltonian cycle in $G_{n,k}$. □

Theorem 6.4. *For the function f_{n_1,n_2} defined above, we have*

$$\operatorname{cnf}(f_{n_1,n_2}) = \binom{n_1}{k} + kn_2 \tag{3}$$

$$\binom{n_1}{k} + n_2 \leq \operatorname{ess}(f_{n_1,n_2}) < 2^{n_1} + n_2. \tag{4}$$

Proof. In order to prove (3), we may restrict ourselves to prime CNFs representing f_{n_1,n_2}. Given an arbitrary prime CNF φ representing f_{n_1,n_2} and an arbitrary $A' \subseteq A$ of size $|A'| = k$, φ has to contain at least one clause of the form $\bigvee_{i \in A'} \overline{x_i} \vee x_j$ where $j \in A \setminus A'$, since otherwise $\mathrm{FC}_\varphi(A') = A'$ contradicting the fact that $\bigvee_{i \in A'} \overline{x_i} \vee x_j$ is an implicate of f_{n_1,n_2} for every $j \in A \setminus A'$. Hence, φ has to contain at least $\binom{n_1}{k}$ clauses of this form. To show that this number of clauses of this type is also sufficient, use Lemma 6.3 to prove the existence of a cycle consisting of all subsets of A of size k and such that sets, which are neighbors in the cycle, have symmetric difference of size 2. Then, consider the cycle as an ordered cycle with any of the two possible orderings. Finally, for each set A' in the cycle, consider the clause $\bigvee_{i \in A'} \overline{x_i} \vee x_j$, where j is the uniquely determined index not contained in A', but contained in the set, which follows A' in the cycle. It is easy to verify that the obtained set of clauses of size $\binom{n_1}{k}$ generates by forward chaining every prime clause listed in item 2 in Lemma 6.1 (by starting with the set of subgoals and following the cycle to the desired head).

Similarly, for each $i \in B$, φ has to contain a superset of the set of clauses $P(i, A')$ for some set A' of size k, since otherwise $\mathrm{FC}_\varphi(\{y_i\})$ contains only those x_j for which clauses $\overline{y_i} \vee x_j$ are explicitly present in φ, contradicting the fact that $\overline{y_i} \vee x_j$ is an implicate of f_{n_1,n_2} for every $j \in A$. Thus φ contains at least kn_2 clauses of this form. To show that this number of clauses of this type is also sufficient, take for each $i \in B$ exactly one set $P(i, A')$ for some arbitrary set A' of size k. Now $\mathrm{FC}_\varphi(\{y_i\})$ contains all x_j for $j \in A'$ and using the cycle of clauses from the previous paragraph forward chaining derives all remaining x_j for $j \in A \setminus A'$.

In order to prove (4), we use Theorem 6.2 to find disjoint essential sets. Let us consider the two types of minimal essential sets listed in Theorem 6.2 separately. For each index $i \in B$, we can choose at most one of the sets $P(i, A')$, since for every pair of sets A', A'', sets $P(i, A')$ and $P(i, A'')$ have nonempty intersection. Hence, we have at most n_2 disjoint minimal essential sets of this type. If we choose $Q(A')$ for all $A' \subseteq A$ of size $|A'| = k$, we obtain $\binom{n_1}{k}$ further disjoint essential sets. On the other hand, the number of different sets A' is at most 2^{n_1}. Altogether, we can find at least $\binom{n_1}{k} + n_2$ and at most $2^{n_1} + n_2$ pairwise disjoint minimal essential sets. □

Corollary 6.5. *For fixed n_1, k and $n_2 \to \infty$, we have $\mathrm{cnf}(f_{n_1,n_2})/\mathrm{ess}(f_{n_1,n_2}) \to k$.*

7. Hardness of computing ess(*f*) for pure Horn 3CNFs

In this section we shall show that the following problem is *NP*-complete:

Problem: *ESS-Horn-3CNF.*
Input: A pure Horn 3CNF φ representing a pure Horn function f and an integer $k \geq 0$.
Question: Is $\mathrm{ess}(f) \geq k$?

We shall prove the hardness of this problem by a transformation from the problem of finding a maximum independent set in a graph G. This reduction is inspired by a similar construction in [2] where a reduction from the Set Cover problem to Boolean minimization (BM) is presented. For this purpose, let us associate a pure Horn function f_G with every undirected graph $G = (V, E)$, where $V = \{x_1, x_2, \dots, x_n\}$, $n = |V|$, $E = \{e_{i,j} \mid e_{i,j} = \{x_i, x_j\}\}$, and $m = |E|$. With every vertex $x_i \in V$ of G we associate a Boolean variable x_i and similarly with every edge $e_{i,j} \in E$ we associate a Boolean variable $e_{i,j}$ (note that since G is an undirected graph, $e_{i,j} = e_{j,i}$). f_G is then a function on $n + m + 1$ variables, n variables associated with vertices, m variables associated with edges and an additional variable z. f_G is defined by the following pure Horn CNF expression

$$\mathcal{F}_G = \bigwedge_{e_{i,j} \in E} \Big((\overline{e_{i,j}} \vee x_i) \wedge (\overline{e_{i,j}} \vee x_j) \wedge (\overline{x_i} \vee \overline{x_j} \vee e_{i,j}) \Big) \wedge \bigwedge_{i=1}^{n} (\overline{z} \vee x_i).$$

Let us at first examine, how the prime implicates of f_G (i.e. the set $\mathcal{I}^p(f_G)$) may look like.

Lemma 7.1. *Let G be an arbitrary undirected graph and let f_G be its associated pure Horn function defined by CNF $\mathcal{F}_G$. A clause C is a prime implicate of f_G if and only if one of the following is true:*

(a) *$C = (\overline{e_{i,j}} \vee x_i)$ for some edge $e_{i,j} \in E$,*
(b) *$C = (\overline{x_i} \vee \overline{x_j} \vee e_{i,j})$ for some edge $e_{i,j} \in E$,*
(c) *$C = (\overline{e_{i,j}} \vee \overline{x_k} \vee e_{i,k})$ for some edges $e_{i,j}, e_{i,k} \in E$ where x_i, x_j, and x_k are three pairwise different vertices of G,*
(d) *$C = (\overline{e_{i,j}} \vee \overline{e_{k,l}} \vee e_{i,k})$ for some edges $e_{i,j}, e_{i,k}, e_{k,l} \in E$ where x_l may be the same vertex as x_j,*
(e) *$C = (\overline{z} \vee e_{i,j})$ for some edge $e_{i,j} \in E$, or*
(f) *$C = (\overline{z} \vee x_i)$ for some vertex $x_i \in V$.*

Proof. Let us first verify that each clause described in the proposition of the lemma is an implicate of f_G. The cases (a), (b), and (f) are trivial as these are the clauses appearing directly in $\mathcal{F}_G$. A clause $C = (\overline{e_{i,j}} \vee \overline{x_k} \vee e_{i,k})$ from case (c) is a resolvent of $C_1 = (\overline{e_{i,j}} \vee x_i)$ and $C_2 = (\overline{x_i} \vee \overline{x_k} \vee e_{i,k})$ and is therefore an implicate of f_G. A clause $C = (\overline{e_{i,j}} \vee \overline{e_{k,l}} \vee e_{i,k})$ from case (d) is a resolvent of $C_1 = (\overline{e_{i,j}} \vee \overline{x_k} \vee e_{i,k})$, which is an implicate due to (c), and $C_2 = (\overline{e_{k,l}} \vee x_k)$, which is an implicate due to (a). A clause $C = (\overline{z} \vee e_{i,j})$ from case (e) is a resolvent of $C_1 = (\overline{z} \vee x_i)$ and $C_2 = (\overline{z} \vee \overline{x_i} \vee e_{i,j})$, where C_2 is a resolvent of $C_4 = (\overline{z} \vee x_j)$ and $C_5 = (\overline{x_i} \vee \overline{x_j} \vee e_{i,j})$.

In order to verify that the clauses (a)–(f) are prime implicates, let us use the following set of satisfying assignments.

- All ones assignment.
- For every vertex x_a, $a \in \{1, \dots, n\}$, the assignment, which sets z, x_a and $e_{a,i}$ for all $i \neq a$ to 0 and all other variables to 1.

Consider any of the clauses (a)–(f). For every literal in it, it is possible to find an assignment from the above list, which satisfies the chosen literal, but no other literal in the considered clause. Since the assignment satisfies the whole formula, the literal cannot be omitted without changing the represented function. This implies that the clauses (a)–(f) are prime implicates of f_G.

For the other direction let us start by examining the forward chaining closure of a set of variables S with respect to f_G which shall be denoted by $\mathrm{FC}_G(S)$. Given an arbitrary set S of variables of f_G, let us denote by $V_S = \{x_i \mid x_i \in S \text{ or } x_i \in e_{i,k} \text{ for some } e_{i,k} \in S\}$, i.e. V_S consists of those vertices which are either present in S directly, or they are incident to some edge, which belongs to S. By E_S let us denote the set of edges of G, whose both vertices belong to V_S. Now we claim that

$$\mathrm{FC}_G(S) = \begin{cases} V \cup E \cup \{z\} & \text{if } z \in S \\ V_S \cup E_S & \text{otherwise.} \end{cases}$$

Let us at first assume that $z \in S$. Then according to the fact that clauses in (e) and (f) are implicates of f_G, we can derive everything from z and therefore clearly $\mathrm{FC}_G(\{z\}) = \mathrm{FC}_G(S) = V \cup E \cup \{z\}$. Now let us assume that $z \notin S$. By using clauses from (a) and (b) we can observe that $V_S \cup E_S \subseteq \mathrm{FC}_G(S)$. By definition of $V_S \cup E_S$ we can see that $S \subseteq V_S \cup E_S$. Let C be a clause in $\mathcal{F}_G$ and let us assume that all its subgoals are contained in $V_S \cup E_S$. We shall show that in this case also its head belongs to $V_S \cup E_S$, and thus it follows that $\mathrm{FC}_G(S) = V_S \cup E_S$. Let us at first assume that C is of type (a), i.e. $C = (\overline{e_{i,j}} \vee x_i)$. In this case $e_{i,j} \in E_S$ and therefore by definition of E_S, we have that $x_i \in V_S$. Now, let us assume that C is of type (b), i.e. $C = (\overline{x_i} \vee \overline{x_j} \vee e_{i,j})$ for some edge $e_{i,j} \in E$. In this case $x_i, x_j \in V_S$ and hence also $e_{i,j} \in E_S$.

Now let us assume that $C = (S \vee y)$ is an implicate of f_G, which is nontrivial, i.e. $y \notin S$. If $z \in S$, then z itself is sufficient for deriving anything and therefore if C should be prime, then $S = \{z\}$, $y \in V \cup E$ and C has the form of (e) or (f). If $z \notin S$, then since C is an implicate of f_G, we get that $y \in \mathrm{FC}_G(S) \subseteq V_S \cup E_S$. If $y = x_i \in V_S$, then since $y \notin S$ it must be the case that $e_{i,j} \in S$ for some edge $e_{i,j} \in E$ incident to x_i. If C should be prime, then we must have $S = \{e_{i,j}\}$ and C has the form of (a). If on the other hand $y = e_{i,k} \in E_S$, then we have three possibilities.

1. $x_i, x_k \in S$. In this case, if C is prime, then $S = \{x_i, x_k\}$ and C has the form of (b).
2. $e_{i,j}, x_k \in S$ for some $e_{i,j} \in E$ or $e_{j,k}, x_i \in S$ for some $e_{j,k} \in E$. In the former case, if C is prime, we have that $S = \{e_{i,j}, x_k\}$ and C has the form of (c). The latter case is symmetric.
3. $e_{i,j}, e_{k,l} \in S$ for some $e_{i,j}, e_{k,l} \in E$ and then if C is prime, we must have that $S = \{e_{i,j}, e_{k,l}\}$ and C has the form of (d).

By this we have shown that every prime implicate of f_G must have the form of one of the cases (a)–(f) in the proposition of the lemma. □

Let us denote the size of the largest independent set of the undirected graph G by $\alpha(G)$, then we claim that the following holds.

Theorem 7.2. *Let $G = (V, E)$ be an undirected graph, then $\alpha(G) = \mathrm{ess}(f_G) - 3m$, where $m = |E|$.*

Proof. Let us at first assume that we have an independent set I of G of size $\alpha(G)$. We shall define three sets of $(m+n+1)$-bit vectors, which define pairwise disjoint essential sets.

1. Given an edge $e_{i,j}$ and a vertex $x_i \in e_{i,j}$ we define vector $t^i_{i,j}$
   - $t^i_{i,j}[x_j] = 1$,
   - $t^i_{i,j}[x_k] = 0$ for $x_k \in V \setminus \{x_j\}$,
   - $t^i_{i,j}[e_{i,j}] = 1$,
   - $t^i_{i,j}[e_{k,l}] = 0$ for $e_{k,l} \in E \setminus \{e_{i,j}\}$, and
   - $t^i_{i,j}[z] = 0$.
2. Given an edge $e_{i,j} \in E$, we define vector $t_{i,j}$
   - $t_{i,j}[x_i] = t_{i,j}[x_j] = 1$,
   - $t_{i,j}[x_k] = 0$ for $x_k \in V \setminus \{x_i, x_j\}$,
   - $t_{i,j}[e_{k,l}] = 0$ for $e_{k,l} \in E$ (including $e_{i,j}$), and
   - $t_{i,j}[z] = 0$.
3. Given $x_a \in I$ we define vector t_a as follows.
   - $t_a[x_a] = 0$,
   - $t_a[x_j] = 1$ for $x_j \in V \setminus \{x_a\}$,
   - $t_a[e_{a,k}] = 0$ for $e_{a,k} \in E$,
   - $t_a[e_{j,k}] = 1$ for $e_{j,k} \in E$ where $x_a \notin e_{j,k}$, and
   - $t_a[z] = 1$.

Let us prove that the prime essential sets defined using these falsepoints are

$$\mathcal{E}^p(t^i_{i,j}) = \{(\overline{e_{i,j}} \vee x_i)\}, \tag{5}$$

$$\mathcal{E}^p(t_{i,j}) = \{(\overline{x_i} \vee \overline{x_j} \vee e_{i,j})\}, \quad \text{and} \tag{6}$$

$$\mathcal{E}^p(t_a) = \{(\overline{z} \vee x_a)\} \cup \{(\overline{z} \vee e_{a,j}) \mid e_{a,j} \in E\} \tag{7}$$

and, in particular, these essential sets are disjoint.

- Let us at first consider vector $t^i_{i,j}$ for an arbitrary edge $e_{i,j} \in E$. Clearly $(\overline{e_{i,j}} \vee x_i) \in \mathcal{E}^p(t^i_{i,j})$. On the other hand, let C be an implicate of f_G for which $C(t^i_{i,j}) = 0$, then the subgoals of C may contain only x_j and $e_{i,j}$ (because these are the only bits set to 1 in $t^i_{i,j}$). According to Lemma 7.1, this condition is satisfied by the prime implicates $(\overline{e_{i,j}} \vee x_j)$ and $(\overline{e_{i,j}} \vee x_i)$. The former is not falsified by $t^i_{i,j}$, therefore the latter is the only prime implicate which belongs to $\mathcal{E}^p(t^i_{i,j})$.
- Now let us consider vector $t_{i,j}$ for an arbitrary $e_{i,j} \in E$. Clearly $(\overline{x_i} \vee \overline{x_j} \vee e_{i,j}) \in \mathcal{E}^p(t_{i,j})$. On the other hand, let C be an implicate of f_G, for which $C(t_{i,j}) = 0$, then the subgoals of C may contain only x_i and x_j (because these are the only bits set to 1 by $t_{i,j}$), According to Lemma 7.1, the only prime implicate satisfying this condition is $(\overline{x_i} \vee \overline{x_j} \vee e_{i,j})$ and it is also the only implicate belonging to $\mathcal{E}^p(t_{i,j})$.

- Now let us consider an arbitrary $x_a \in I$ and the corresponding vector t_a. The fact that $\{(\overline{z} \vee x_a)\} \cup \{(\overline{z} \vee e_{a,j}) \mid e_{a,j} \in E\} \subseteq \mathcal{E}^p(t_a)$ follows from the definition of t_a and it should also be clear that every clause with subgoal z, which belongs to $\mathcal{E}^p(t_a)$, is contained in the left-hand side of the above set inclusion. It therefore remains to show that every implicate C, which does not contain z as a subgoal, evaluates to 1 on t_a. Since C does not contain z as a subgoal, it must have the form (a)–(c), or (d) from the proposition of Lemma 7.1. If $C = (\overline{e_{i,j}} \vee x_i)$ for some $e_{i,j} \in E$ (case (a)), then $C(t_a) = 1$ because if $t_a[x_i] = 0$, then $i = a$ and $t_a[e_{i,j}] = 0$. If $C = (\overline{x_i} \vee \overline{x_j} \vee e_{i,j})$ for some edge $e_{i,j} \in E$ (case (b)), then $C(t_a) = 1$, because if x_i and x_j are both set to 1 by t_a, then also $e_{i,j}$ is set to 1. If $C = (\overline{e_{i,j}} \vee \overline{x_k} \vee e_{i,k})$ for some edges $e_{i,j}, e_{i,k}$ (case (c)), then either $a \in \{i, j, k\}$ in which case one of $e_{i,j}$ or x_k is set to 0 by t_a, or $a \notin \{i, j, k\}$ and $t_a[e_{i,k}] = 1$, therefore also in this case $C(t_a) = 1$. If $C = (\overline{e_{i,j}} \vee \overline{e_{k,l}} \vee e_{i,k})$ for some edges $e_{i,j}, e_{k,l}, e_{i,k} \in E$ (case (d)), then either $a \in \{i, j, k, l\}$, in which case $t_a[e_{i,j}] = 0$ or $t_a[e_{k,l}] = 0$, or $a \notin \{i, j, k, l\}$, in which case $t_a[e_{i,k}] = 1$. According to Lemma 7.1, there are no other prime clauses, which could belong to $\mathcal{E}^p(t_a)$.

The above shows that the sets of type $\mathcal{E}(t^i_{i,j})$ and $\mathcal{E}(t_{i,j})$ are pairwise disjoint and that they are disjoint with sets of type $\mathcal{E}(t_a)$. It remains to show that given two different $x_a, x_b \in I$, the sets $\mathcal{E}(t_a)$ and $\mathcal{E}(t_b)$ are disjoint. A clause $(\overline{z} \vee x_a) \notin \mathcal{E}(t_b)$ and similarly $(\overline{z} \vee x_b) \notin \mathcal{E}(t_a)$, therefore if there is a clause in $\mathcal{E}(t_a) \cap \mathcal{E}(t_b)$, then it is the clause $(\overline{z} \vee e_{a,b})$. However, this clause is not a prime implicate by Lemma 7.1, because I is an independent set and hence $\{x_a, x_b\} \notin E$, which means that $e_{a,b}$ does not appear as a variable in $\mathcal{F}_G$. The number of pairwise disjoint essential sets we have found is $3|E| + |I| = 3m + \alpha(G)$. This implies that $\text{ess}(f_G) \geq 3m + \alpha(G)$.

Now let us assume that we have $\text{ess}(f_G)$ pairwise disjoint FE sets of f_G. The above construction implies that $\text{ess}(f_G) \geq 3m$. We shall construct an independent set I of size $k = \text{ess}(f_G) - 3m \geq 0$. Let the falsepoints defining the $\text{ess}(f_G)$ pairwise disjoint sets be denoted by $s_1, \ldots, s_{\text{ess}(f_G)}$ and assume without loss of generality that each $\mathcal{E}^p(s_i)$ is minimal. Prime implicates C, which have the form (a) or (b) of the proposition of Lemma 7.1, form themselves singleton prime essential sets, which are also minimal essential sets. Hence, for every such C, we have some i for which $\mathcal{E}^p(s_i) = \{C\}$, otherwise, we could find a larger collection of pairwise disjoint essential sets by adding $\{C\}$ to it. Let us assume that $s_1, \ldots, s_{3m}$ correspond to these singleton sets. Therefore, $s_{3m+1}, \ldots, s_{3m+k}$ are falsepoints which define the remaining k essential sets. Let us inspect the set $\mathcal{E}^p(s_{3m+w})$ for an arbitrary $w \in \{1, \ldots, k\}$. The set $\mathcal{E}^p(s_{3m+w})$ must have a nonempty intersection with $\mathcal{F}_G$ by Theorem 5.8, but it cannot contain clauses of the form (a) or (b) from the proposition of Lemma 7.1, because $\mathcal{E}^p(s_{3m+w})$ is disjoint with $\mathcal{E}^p(s_i)$ for every $i \in \{1, \ldots, 3m\}$. Hence, it must contain a clause $(\overline{z} \vee x_i)$ for some variable $x_i \in V$. Let us associate with every vector s_{3m+w} one of these variables and let us denote it by x_{i_w}. We set $I = \{x_{i_1}, \ldots, x_{i_k}\}$ and claim that I is an independent set of G. Let $w, y \in \{1, \ldots, k\}$ be two arbitrary, but distinct indices and x_{i_w}, x_{i_y} their associated variables. Let us assume by contradiction that $e_{i_w,i_y} = \{x_{i_w}, x_{i_y}\} \in E$. Since $(\overline{z} \vee x_{i_w}) \in \mathcal{E}^p(s_{3m+w})$ and $(\overline{z} \vee x_{i_y}) \in \mathcal{E}^p(s_{3m+y})$, we have $s_{3m+w}[z] = s_{3m+y}[z] = 1$ and $s_{3m+w}[x_{i_w}] = s_{3m+y}[x_{i_y}] = 0$. Since $\mathcal{E}^p(s_{3m+w}) \cap \mathcal{E}^p(s_{3m+y}) = \emptyset$, we also have $s_{3m+w}[x_{i_y}] = s_{3m+y}[x_{i_w}] = 1$. Moreover, at least one of $s_{3m+w}[e_{i_w,i_y}]$ and $s_{3m+y}[e_{i_w,i_y}]$ must be 1, otherwise $(\overline{z} \vee e_{i_w,i_y}) \in \mathcal{E}^p(s_{3m+w}) \cap \mathcal{E}^p(s_{3m+y})$. But if $s_{3m+y}[e_{i_w,i_y}] = 1$, then $(\overline{e_{i_w,i_y}} \vee x_{i_y})$ evaluates to 0 on s_{3m+y}, which is a contradiction to the disjointness of $\mathcal{E}^p(s_{3m+y})$ and $\{(\overline{e_{i_w,i_y}} \vee x_{i_y})\}$ included among $\mathcal{E}^p(s_1), \ldots, \mathcal{E}^p(s_{3m})$. Similarly, if $s_{3m+w}[e_{i_w,i_y}] = 1$, then $(\overline{e_{i_w,i_y}} \vee x_{i_w})$ evaluates to 0 on s_{3m+w}, which is a contradiction to the disjointness of $\mathcal{E}^p(s_{3m+w})$ and $\{(\overline{e_{i_w,i_y}} \vee x_{i_w})\}$. Therefore x_{i_w}, x_{i_y} cannot form an edge of G. By this we have shown that I is an independent set of G of size $|I| = k \leq \alpha(G)$ and, hence, $\text{ess}(f_G) = 3m + k \leq 3m + \alpha(G)$.

The first and the second half of the proof together imply that $\text{ess}(f_G) = 3m + \alpha(G)$. □

The fact that the problem ESS-Horn-3CNF belongs to *NP* follows directly from Theorem 3.8 and we may therefore conclude the following.

Corollary 7.3. *The problem ESS-Horn-3CNF is NP-complete.*

It is also worth to note that while computing $\text{ess}(f_G)$ is NP-hard (equivalent to computing $\alpha(G)$), computing $\text{cnf}(f_G)$ can be done in polynomial time. As was shown in [2], computing $\text{cnf}(f_G)$ is equivalent to computing the size of a minimum edge cover of G, which is long known to be in P.

8. Computing ess(*f*) and its relaxation from the truth table of *f*

In this section we first prove *NP*-completeness of the following problem.

Problem *ESS-TT*(f, k).
Input: A Boolean function f represented by its truth table and an integer $k \geq 0$.
Question: Is $\text{ess}(f) \geq k$?

Minimization of DNF for a Boolean function given by its truth table is proved to be *NP*-hard in [1] using a reduction from 3-*Partite Set Cover*. We use essentially the same reduction, although we use it as a reduction of the problem 3-*PARTITE-TRIANG-INDSET* described below to *ESS-TT*. An instance of the input problem is a 3-uniform hypergraph $\mathcal{H} = (V, S)$, whose set of edges S is a subset of $U_1 \times U_2 \times U_3$ for some pairwise disjoint sets of vertices U_1, U_2, and U_3. We consider this hypergraph as a representation of an ordinary graph, which is a union of a set of 3-partite triangles. Namely, for $\mathcal{H}$ as above, we define $\mathcal{G}(\mathcal{H})$ as a graph on the set of vertices V and whose edges are all two-element subsets of the edges in S. Formally, we define problem 3-*PARTITE-TRIANG-INDSET*$(\mathcal{H}, k)$ as follows:

Problem 3-*PARTITE-TRIANG-INDSET*($\mathcal{H}, k$).
Input: A hypergraph $\mathcal{H} = (U, \mathcal{S})$, where $U = U_1 \cup U_2 \cup U_3$ and $\mathcal{S} \subseteq U_1 \times U_2 \times U_3$ for some pairwise disjoint sets of vertices U_1, U_2, and U_3, and an integer $k \geq 0$.
Question: Is there an independent set of vertices $I \subseteq U$ in $\mathcal{G}(\mathcal{H})$, $|I| \geq k$?

Note that the instance of 3-*PARTITE-TRIANG-INDSET* is the same as in 3-*Partite Set Cover*, the only difference is in the question, which we ask about the input hypergraph. We shall start by proving that the problem we have just defined is *NP*-complete.

Theorem 8.1. 3 − PARTITE − TRIANG − INDSET*($\mathcal{H}, k$) problem is NP-complete.*

Proof. The problem is clearly in *NP*, since an independent set I of at most k vertices can serve as a polynomially verifiable certificate of a positive answer. We prove that it is *NP*-complete using a reduction from the maximum independent set problem restricted to instances $(G = (V, E), k)$, where G is a graph with no isolated vertices satisfying $|E| \geq |V|$ and $k \geq 2$. In order to see that this problem is *NP*-complete, consider an unrestricted instance of maximum independent set problem. If we eliminate all isolated vertices and decrease the size bound for the independent set accordingly, we get an equivalent instance. If we add a vertex connected to all vertices of the original graph, we do not change the size of the maximum independent set, but we get a graph, which has at least so many edges as vertices. The assumption that $k \geq 2$ does not change the *NP*-complete status of the problem, since the instances with $k \leq 1$ are trivial.

In order to reduce the problem from the previous paragraph into 3-*PARTITE-TRIANG-INDSET* problem, we construct a 3-partite graph G', using a reduction from [16]. Namely, construct $G' = (V', E')$ from G by replacing each edge by a path of length 3 and consider the instance $(G', k + |E|)$ of the maximum independent set problem. Note that V' consists of the original vertices and of $2|E|$ new vertices, which are internal vertices of the paths replacing original edges. Let V_2 be a set of the new nodes containing one of the two nodes from each of these paths chosen arbitrarily. Let V_3 be the set of the remaining new nodes. For simplicity of notation, let us denote $V_1 = V$. Then, we have $V' = V_1 \cup V_2 \cup V_3$ and these three sets form the partitions of the 3-partite graph G'. Since $|V| \leq |E|$, we have $|V_1| \leq |V_2| = |V_3| = |E|$.

Using the argument from [16], G contains an independent set of size k if and only if G' contains an independent set of size $k + |E|$.

Let $G'' = (V'', E'')$ be obtained from G' by adding three nodes u_1, u_2, u_3 and all edges between u_i and all vertices of V_j, where $i \neq j$. Note that G'' is 3-partite with the partitions $V_i \cup \{u_i\}$ for $i = 1, 2, 3$. Consider the instance of maximal independent set $(G'', k + |E|)$. We will prove that it is equivalent to the instance $(G', k + |E|)$. The independent sets of G'' are of two types. If it contains one of the nodes u_i, then it is a subset of $V_i \cup \{u_i\}$. If it does not contain any of the nodes u_i, then it is an independent set in G'. Now note that the independent sets contained in $V_i \cup \{u_i\}$ have size smaller than $k + |E|$, since $|V_1| \leq |V_2| = |V_3| = |E|$ and $k \geq 2$.

Given 3-partite graph G'', it is easy to construct a hypergraph $\mathcal{H}$ such that $G'' = \mathcal{G}(\mathcal{H})$. Namely, let

$$U_i = V_i \cup \{u_i\}, \quad i = 1, 2, 3,$$

and

$$\mathcal{S} = \{\{v_i, v_j, u_k\} \mid \{v_i, v_j\} \in E' \text{ and } \{i, j, k\} = \{1, 2, 3\}\}.$$

The sets U_i are clearly disjoint. Since, $G'' = \mathcal{G}(\mathcal{H})$, the instance $(G'', k + |E|)$ of maximum independent set is equivalent to the instance $(\mathcal{H}, k + |E|)$ of 3-*PARTITE-TRIANG-INDSET*. □

Now we shall describe a reduction of an instance $(\mathcal{H}, k)$ of 3-*PARTITE-TRIANG-INDSET* to an instance of *ESS-TT*, i.e. to a Boolean function f. The construction we use here is the same as the one used in [1] to transform an instance of 3-Partite Set Cover to a minimization of DNF for a Boolean function given by its truth table. Here, we use the fact that an instance of 3-*PARTITE-TRIANG-INDSET* is described by the same hypergraph as an instance of 3-Partite Set Cover, use the same transformation to a Boolean function and consider its negation, since we are dealing with CNFs instead of DNFs. However, finally, we ask a different question about the constructed Boolean function, which is equivalent to a question on the input instance considered as a 3-*PARTITE-TRIANG-INDSET* and not a 3-Partite Set Cover instance. Since the proof of the correctness of the transformation for our purpose is different from the one in [1], we describe the transformation here in full detail.

Let $\mathcal{H} = (U_1 \cup U_2 \cup U_3, \mathcal{S})$ be an arbitrary hypergraph, where $\mathcal{S} \subseteq U_1 \times U_2 \times U_3$ for some pairwise disjoint sets of vertices U_1, U_2, and U_3. We shall describe, how to associate a Boolean function $f_{\mathcal{H}}$ with $\mathcal{H}$ in the same way as it was done in [1].

Let $n = \max\{|U_1|, |U_2|, |U_3|\}$, let q be the smallest integer satisfying $\binom{q}{q/2} \geq n$, and let $t = 3q$. Note that the fact that q is the smallest integer with the required property implies that $q = O(\log n)$. Let $b(j)$ for $j = 1, \ldots, n$ be distinct vectors from $\{0, 1\}^q$ each of which contains exactly $q/2$ ones. Then, let $V \subseteq \{0, 1\}^t$ be such that it contains encodings of the elements of $U_1 \cup U_2 \cup U_3$ defined as follows. The jth element u of U_i, where $j = 1, \ldots, |U_i|$, is encoded by a vector $e(u)$ consisting of three blocks of length q, ith of which is $b(j)$ and the remaining two blocks consist of q zeros.

For each $A \in \mathcal{S}$, let its encoding $e(A)$ be the bitwise disjunction of the encodings of the three elements of A in V. Note that the construction of the encodings guarantees that different sets A correspond to incomparable vectors in $\{0, 1\}^t$, since the sets differ in at least one of their elements and the corresponding blocks of length q are incomparable. Let $W = \{e(A) \mid A \in \mathcal{S}\}$. The following lemma was proved in [1].

Lemma 8.2 *([1], First Part of Lemma 3.1).* *For each $A \in \mathcal{S}$ and each $u \in U_1 \cup U_2 \cup U_3$, we have*

$$u \in A \Leftrightarrow e(u) \leq e(A).$$

Let $R = \{x \in \{0,1\}^t \mid x \notin V \text{ and for some } w \in W, x \leq w\}$. Let g be a partial function with the domain $\{0,1\}^t$ such that $g(x) = 0$ if $x \in V$, $g(x) = *$ if $x \in R$, and $g(x) = 1$ otherwise. We shall finish the transformation by reduction of the partial function g of the variables $x \in \{0,1\}^t$ to the total function $f_{\mathcal{H}}$ of the variables $(x, y_1, y_2) \in \{0,1\}^{t+2}$

$$f_{\mathcal{H}}(x, y_1, y_2) = \begin{cases} 0, & \text{if } g(x) = 0 \text{ and } y_1 = y_2 = 1 \\ 0, & \text{if } g(x) = * \text{ and } y_1 = y_2 = 1 \\ 0, & \text{if } g(x) = * \text{ and } y_1 = p(x), \text{ and } y_2 = \neg p(x) \\ 1, & \text{otherwise} \end{cases}$$

where $p(x)$ is the parity of x, i.e. the sum of the bits in x mod 2.

As we have already mentioned, the construction of the function $f_{\mathcal{H}}$ is exactly the same as described in [1] (see also [7]) with the only difference caused by using CNFs instead of DNFs—we had to negate the final function $f_{\mathcal{H}}$. Now we shall show that $\text{ess}(f_{\mathcal{H}}) \geq |R| + k$ if and only if the graph $\mathcal{G}(\mathcal{H})$ has an independent set of size k.

Theorem 8.3. *Let $\mathcal{H} = (U = U_1 \cup U_2 \cup U_3, \mathcal{S} \subseteq U_1 \times U_2 \times U_3)$, where U_1, U_2, U_3 are pairwise disjoint sets of vertices, let $f_{\mathcal{H}}$ be its associated Boolean function, let R be the set of inputs defined during its construction, and let k be an arbitrary integer. Then $\text{ess}(f_{\mathcal{H}}) \geq |R| + k$ if and only if the hypergraph $\mathcal{H}$ has an independent set of size k.*

Proof. Let us start by description of the list of all prime implicates of $f_{\mathcal{H}}$. For every $x \in R$, consider the clause, which is 0 on the two points $(x, p(x), \neg p(x))$ and $(x, 1, 1)$. This is a prime implicate, since $(x, 1, 1)$ is the only neighbor of $(x, p(x), \neg p(x))$, where $f_{\mathcal{H}}$ is 0. Moreover, these are the only prime implicates, which are zero on some of the points with $y_1 = 0$ or $y_2 = 0$. Hence, all the remaining prime implicates are zero only on the points $(z, 1, 1)$ for some $z \in \{0,1\}^t$. Since $f_{\mathcal{H}}(z, 1, 1) = 0$ if and only if $z \leq w$ for some $w \in W$ and the elements of W are pairwise incomparable, it is easy to verify that all the remaining prime implicates of $f_{\mathcal{H}}$ may be obtained in such a way that for any $w \in W$, we consider the clause, which is 0 exactly on the vectors $(z, 1, 1)$, where $z \leq w$ for the given w.

Now, assume that I is an independent set in $\mathcal{G}(\mathcal{H})$ of size k and let us construct a set of $|R| + k$ essential sets of $f_{\mathcal{H}}$, which are pairwise disjoint. Consider the prime essential sets $\mathcal{E}^p((x, p(x), \neg p(x)))$ for $x \in R$ and $\mathcal{E}^p(e(v))$ for $v \in I$. Prime essential sets of the former type contain a single clause, which is not contained in any other prime essential set from the presented list. Hence, these essential sets are disjoint with all the others. Consider prime essential sets $\mathcal{E}^p(e(v_1))$ and $\mathcal{E}^p(e(v_2))$ for different points $v_1, v_2 \in I$. If these two essential sets are not disjoint, then there is a vector $w \in W$ such that $v_1 \leq w$ and $v_2 \leq w$. By the definition of W, this implies that there is $A \in S$ such that $v_1, v_2 \in A$ and so, (v_1, v_2) is an edge of $\mathcal{G}(\mathcal{H})$. This is not possible, since I is an independent set. Hence, the presented $|R| + k$ essential sets are indeed disjoint.

For the opposite direction, assume that Z is a set of vectors in $\{0,1\}^{t+2}$ such that $|Z| \geq |R| + k$ and the prime essential sets $\mathcal{E}^p(z)$ for all $z \in Z$ are pairwise disjoint. If some of the points $(x, p(x), \neg p(x))$ is not in Z, then modify Z by including this point to Z and removing the point $(x, 1, 1)$ from Z, if it is there. The size of Z does not decrease and one may verify that the points from the modified Z still define disjoint prime essential sets. Now, note that except of $|R|$ points $(x, p(x), \neg p(x))$, Z contains only points from V. Hence, there are at least k points $v \in V$, such that for every $w \in W$, at most one of them satisfies $v \leq w$. These k vectors from V are encodings of k vertices of $\mathcal{H}$, no two of which belong to the same set $A \in \mathcal{S}$. It follows that no two of these points are connected by an edge in $\mathcal{G}(\mathcal{H})$ and so, $\mathcal{G}(\mathcal{H})$ contains an independent set of size k. □

As a corollary we now obtain the following.

Theorem 8.4. *The problem to determine, whether* $\text{ess}(f) \geq k$ *for a function f defined by its truth table, i.e. the problem* ESS − TT*(f, k), is NP-complete.*

Proof. The fact that *ESS-TT* belongs to *NP* can be easily observed and it also follows from Theorem 3.8. *NP*-hardness of *ESS-TT* follows from Theorem 8.1, construction of $f_{\mathcal{H}}$ described in this section, and Theorem 8.3. □

8.1. Relaxation of ess(f) *for functions given by their truth table*

Computing $\text{ess}(f)$ for functions given by their truth table is intractable by Theorem 8.4. On the other hand, it appears that a relaxation of $\text{ess}(f)$ may be computed more efficiently under the same conditions, namely that the truth table of f is given as the input.

Definition 8.5. For every Boolean function f, let $lp(f)$ be the maximum of

$$\sum_{x \in \{0,1\}^n : f(x) = 0} w(x),$$

(over all possible choices of weights w) where $w(x)$ is a nonnegative real number assigned to every falsepoint x of f and such that for every prime implicant C of f, the inequality

$$\sum_{x\in\{0,1\}^n : C(x)=0} w(x) \leq 1$$

is satisfied.

Theorem 8.6. *For every Boolean function f, we have*

$$\mathrm{cnf}(f) \geq lp(f) \geq \mathrm{ess}(f).$$

Proof. Let $\mathcal{F}$ be a set of prime implicates, which form a minimal CNF representation and let $w(x)$ be an assignment of the weights, on which the value $lp(f)$ is achieved in Definition 8.5. Since $f = \bigwedge_{C\in\mathcal{F}} C$, we have also

$$\sum_{x\in\{0,1\}^n : f(x)=0} w(x) \leq \sum_{C\in\mathcal{F}} \sum_{x\in\{0,1\}^n : C(x)=0} w(x) \leq |\mathcal{F}|.$$

Since $|\mathcal{F}| = \mathrm{cnf}(f)$ and $\sum_{x\in\{0,1\}^n : f(x)=0} w(x) = lp(f)$, we have $\mathrm{cnf}(f) \geq lp(f)$.

Let $\mathcal{T}$ be a set of $\mathrm{ess}(f)$ falsepoints t, for which the sets $\mathcal{E}(t)$ are pairwise disjoint. Let $w(t) = 1$ for $t \in \mathcal{T}$ and $w(t) = 0$ otherwise. If C is a prime implicate, then it belongs to at most one $\mathcal{E}(t)$, $t \in \mathcal{T}$, and thus there is at most one falsepoint t falsifying C for which $w(x) = 1$. Hence, the weights $w(x)$ satisfy the condition in Definition 8.5. Since $lp(f)$ is a maximum over all possible weights satisfying this condition, we have

$$lp(f) \geq \sum_{x\in\{0,1\}^n : f(x)=0} w(x) = |\mathcal{T}| = \mathrm{ess}(f). \quad \square$$

Let us verify that the linear programming problem corresponding to computing $lp(f)$ has size polynomial in the size of the table of the function f. The variables of the corresponding LP problem are the weights $w(x)$ for all falsepoints of f. If the function has n variables, then the size of the table is 2^n and this is clearly an upper bound on the number of falsepoints. Moreover, the largest set of constraints correspond to prime implicates. Since there are at most $3^n = (2^n)^{\log_2 3}$ clauses on n variables, the number of prime implicates is also bounded by a polynomial in the table size and they can be found in polynomial time. Consequently, the problem can be solved, e.g. by Karmarkar's algorithm [14], in time polynomial in the size of the table of f and the number of bits of the precision of the representation of the numbers used in the computation.

9. Conclusion

In this paper we have studied a lower bound on the minimum CNF size of a given function f represented by a CNF φ. The lower bound which we have considered is given by $\mathrm{ess}(f)$, which denotes the number of pairwise disjoint essential sets of implicates of f. We are mainly interested in functions for which this lower bound matches the minimum CNF size. We have called such functions coverable, and we have shown in Sections 3 and 4 that if a class of Boolean function $\mathcal{C}$ is tractable and coverable, then the problem of minimization of functions in this class belongs to both *NP* and *co-NP*. This fact proves that such a minimization problem is not NP-hard (unless $NP = co\text{-}NP$), and thus indicates that it might in fact belong to *P*. In Section 5 we study the intersections of essential sets with the set of prime implicates, call these intersections prime essential sets and prove that many properties of essential sets carry over to prime essential sets. This fact allows us to restrict our attention to prime implicates only.

We have also proved several negative results about $\mathrm{ess}(f)$. In Section 6 we have shown that not every function is coverable and moreover for every constant k we can construct a function f, for which $\mathrm{cnf}(f)/\mathrm{ess}(f) \geq k$. In Section 7 we have shown that the problem of checking whether $\mathrm{ess}(f) \geq k$ is NP-complete if its input is a pure Horn 3CNF. In Section 8 we have shown that this problem remains NP-complete even in the case when the input is allowed to be much larger, namely when the input function is represented by a truth table. On the other hand we have shown that a relaxed value of $\mathrm{ess}(f)$ can be computed using linear programming.

Given the fact that minimization seems to be easier for coverable functions than in the general case, one might ask, whether it would be possible to check in polynomial time, if a given function f is coverable, i.e. whether $\mathrm{ess}(f) = \mathrm{cnf}(f)$. Unfortunately, it turns out that this problem is NP-complete even if the input is a pure Horn 3CNF. This result is not contained in the present paper, because we have found the corresponding reduction just recently. On the other hand, all classes for which a polynomial time minimization algorithm is known to us are coverable. This gives an indication that if a tractable class of Boolean functions is found to be coverable, then we can hope for a polynomial minimization algorithm for it. From theoretical point of view, it would be therefore interesting to find a class of Boolean functions which would be tractable and coverable, and yet we would not be able to find a polynomial minimization algorithm for it.

Acknowledgments

Petr Savicky was supported by grant number 1M0545 (MŠMT ČR) and by Institutional Research Plan AV0Z10300504. Petr Kučera and Ondřej Čepek gratefully acknowledge the support by the Czech Science Foundation (grants 201/07/P168 and

P202/10/1188). The authors would like to thank the two anonymous referees for their valuable comments which helped to improve the presentation of this paper.

References

[1] E. Allender, L. Hellerstein, P. McCabe, T. Pitassi, M. Saks, Minimizing DNF formulas and AC_d^0 circuits given a truth table, in: Proceedings of the 21st Annual IEEE Conference on Computational Complexity, IEEE Computer Society, 2006, pp. 237–251.

[2] G. Ausiello, A. D'Atri, D. Sacca, Minimal representation of directed hypergraphs, SIAM Journal on Computing 15 (2) (1986) 418–431.

[3] E. Boros, O. Čepek, A. Kogan, P. Kučera, Exclusive and essential sets of implicates of Boolean functions, Discrete Applied Mathematics 158 (2) (2010) 81–96.

[4] H. Kleine Büning, T. Letterman, Propositional Logic: Deduction and Algorithms, Cambridge University Press, New York, NY, 1999.

[5] O. Čepek, P. Kučera, Disjoint essential sets of implicates of a CQ Horn function, in: Proceedings of 12th Czech-Japan Seminar on Data Analysis and Decision Making under Uncertainty, Litomyšl, Czech Republic, 2009, pp. 79–92.

[6] S.A. Cook, The complexity of theorem-proving procedures, in: STOC'71: Proceedings of the Third Annual ACM Symposium on Theory of Computing, New York, NY, 1971, pp. 151–158.

[7] S. Czort, The complexity of minimizing disjunctive normal form formulas, Master's Thesis, University of Aarhus, 1999.

[8] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, San Francisco, CA, 1979.

[9] M.R. Genesereth, N.J. Nilsson, Logical Foundations of Artificial Intelligence, Morgan Kaufmann, Los Altos, CA, 1987.

[10] P.L. Hammer, A. Kogan, Horn functions and their DNFs, Information Processing Letters 44 (1992) 23–29.

[11] P.L. Hammer, A. Kogan, Optimal compression of propositional horn knowledge bases: complexity and approximation, Artificial Intelligence 64 (1993) 131–145.

[12] P.L. Hammer, A. Kogan, Knowledge compression—logic minimization for expert systems, in: Proceedings of IISF/ACM Japan International Symposium, World Scientific, Tokyo, Singapore, 1994, pp. 306–312.

[13] P.L. Hammer, A. Kogan, Quasi-acyclic propositional horn knowledge bases: optimal compression, IEEE Transactions on Knowledge and Data Engineering 7 (5) (1995) 751–762.

[14] N. Karmarkar, A new polynomial time algorithm for linear programming, Combinatorica 4 (4) (1984) 373–395.

[15] D. Maier, Minimal covers in the relational database model, Journal of the ACM 27 (1980) 664–674.

[16] S. Poljak, A note on stable sets and colorings of graphs, Commentationes Mathematicae Universitatis Carolinae 15 (2) (1974) 307–309.

[17] W. Quine, The problem of simplifying the truth functions, American Mathematical Monthly 59 (1952) 521–531.

[18] W. Quine, A way to simplify truth functions, American Mathematical Monthly 62 (1955) 627–631.

[19] S.J. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, Pearson Education, 2003.

[20] C. Umans, The minimum equivalent DNF problem and shortest implicants, Journal of Computer and System Sciences 63 (4) (2001) 597–611.

Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

A decomposition method for CNF minimality proofs

Endre Boros [a], Ondřej Čepek [b], Petr Kučera [b,*]

[a] *RUTCOR, Rutgers University, P.O. Box 5062, New Brunswick, NJ 08903, USA*
[b] *Department of Theoretical Computer Science and Mathematical Logic, Faculty of Mathematics and Physics, Charles University in Prague, Malostranské nám. 25, 118 00 Praha 1, Czech Republic*

A R T I C L E I N F O

Article history:
Received 4 March 2013
Received in revised form 17 July 2013
Accepted 17 September 2013
Communicated by G. Ausiello

Keywords:
Boolean functions
CNF
Boolean minimization
Horn minimization
Decomposition method

A B S T R A C T

A CNF is minimal if no shorter CNF representing the same function exists, where by CNF length we mean either the number of clauses or the total number of literals (sum of clause lengths). In this paper we develop a decomposition approach that can be in certain situations applied to a CNF formula when proving its minimality. We give two examples in which this decomposition approach is used. Both examples deal with pure Horn minimization, a problem defined as follows: given a pure Horn CNF, construct a logically equivalent pure Horn CNF which is the shortest possible (either w.r.t. the number of clauses or w.r.t. the total number of literals). Both presented examples give alternative proofs of known complexity results for pure Horn minimization.

1. Introduction

Boolean functions are often represented by conjunctive normal forms (CNFs). In some applications, an important problem (a so-called Boolean minimization problem) is to find a shortest possible CNF representation of a given Boolean function. For instance, in artificial intelligence this problem is equivalent to finding a most compact representation of a given knowledge base [1,2]. Such transformation of a knowledge base accomplishes knowledge compression, since the actual knowledge does not change, while the size of the representation can be significantly reduced. The procedure of knowledge compression preprocesses the knowledge base, and can be done off-line. This results in speeding up on-line operation while answering queries. Therefore, the computational expense of a single run of knowledge compression may be later amortized over a large number of queries to the knowledge base.

The formal statement of the Boolean minimization problem (BM) depends on how the input function is represented, and how the size of the output CNF formula is measured. In this paper the input to the Boolean minimization problem is assumed to be in a form of a CNF and we consider two possible ways of measuring the size of the output CNF formula, the number of clauses and the total number of literals (sum of clause lengths). It is easy to see that the Boolean minimization problem is NP-hard (for both measures of the size of the output CNF). This is an easy consequence of the fact that BM contains the CNF satisfiability problem (SAT) as its special case (an unsatisfiable formula can be trivially recognized from its shortest CNF representation). BM was shown to be probably harder than SAT: while SAT is NP-complete (i.e. Σ_1^p-complete [3]), BM is Σ_2^p-complete [4] (see also the review paper [5] for related results). It was also shown that BM is Σ_2^p-complete when considering general formulas of constant depth as the input and output to the Boolean minimization problem [6].

It is also long known that BM is NP-hard already for some classes of CNFs where SAT is solvable in polynomial time. Maybe a best known example is the class of Horn CNFs (a CNF is Horn if every clause in it contains at most one positive

[*] Corresponding author. Tel.: +420 221 914 138; fax: +420 221 914 323.
E-mail addresses: Endre.Boros@rutgers.edu (E. Boros), ondrej.cepek@mff.cuni.cz (O. Čepek), kucerap@ktiml.mff.cuni.cz (P. Kučera).

0304-3975/$ – see front matter © 2013 Published by Elsevier B.V.
http://dx.doi.org/10.1016/j.tcs.2013.09.016

literal) where the NP-hardness with respect to both output measures was proved [1,7–10]. There exists a hierarchy of tractable subclasses of Horn CNFs for which there are polynomial time minimization algorithms, namely acyclic and quasi-acyclic Horn CNFs [11], and CQ Horn CNFs [12]. There are also few heuristic minimization algorithms for Horn CNFs [13].

A problem closely related to Boolean minimization is the following: given a CNF, prove that it is the shortest CNF representation of the underlying function. This problem belongs to Π_2^p (for every shorter CNF there exists an assignment of truth values on which the two CNFs differ), its complexity has not yet been fully established, although it is conjectured that it is Π_2^p-complete [6].

In this paper we present a decomposition technique for such CNF minimality proofs. It is based on recent results [14] about certain sets (so-called exclusive sets) of implicates of a Boolean function. The introduced decomposition technique is quite general. It is not specific to any subclass of Boolean functions and, in principle, can be applied to any CNF formula when proving its minimality.

We shall show two examples where the developed decomposition technique is used. Both minimality proofs will be parts of NP-hardness reductions for Horn minimization problems. In these examples, we shall restrict our attention to pure Horn CNFs only (where a CNF is pure Horn if each of its clauses contains exactly one positive literal), omitting negative clauses, which, at least in the case of clause minimization, do not influence the complexity of the problem [1].

The clause minimization problem for pure Horn formulas was first addressed in [7] where its NP-hardness was established. However, the proof is a bit sketchy and in our opinion contains a logical leap which requires a more detailed justification (a detailed discussion of this issue is in Subsection 5.5). Moreover, the reduction in the proof in [7] constructed high degree clauses (with the degree proportional to the number of all variables, where the degree of a clause is the number of literals in it), which left open the question, what is the complexity of clause minimization for pure Horn CNFs of a bounded degree. In [8,9] this question was solved by proving that clause minimization stays NP-hard even for cubic pure Horn CNFs. Unfortunately, as pointed out in [15], the proof contains an error which is probably quite difficult to correct.

The literal minimization results for pure Horn formulas are even older than clause minimization ones. The first NP-hardness proof for the problem appeared in [10]. Although strictly speaking the measure defined in [10] is slightly different from the one used here, the proof can be easily modified to work also for the number of literals. A simpler proof (this time really using the number of literals as the minimality measure) then appeared in [1]. Both of these proofs had the same drawback as the clause minimization proof in [7]. Again, both reductions constructed clauses of a very high degree. The natural question, what is the complexity of literal minimization for pure Horn CNFs of a bounded degree was addressed in [9] where it was proved, that it stays NP-hard when we restrict the input to CNFs of degree at most seven. This was later improved to degree at most five in [16].

Recently, it was shown in [17,18] that pure Horn minimization is not only hard to solve exactly but even hard to approximate. More precisely, [17] shows that this problem is inapproximable within a factor $2^{\log^{1-\varepsilon}(n)}$ assuming $NP \subsetneq DTIME(n^{polylog(n)})$, and [18] that it is inapproximable within a factor $2^{O(\log^{1-o(1)} n)}$ assuming $P \subsetneq NP$ even when the input is restricted to 3-CNFs with $O(n^{1+\varepsilon})$ clauses, for some small $\varepsilon > 0$. The latter result of course implies, that pure Horn minimization (both with respect to the number of clauses and the number of literals) is NP-hard already for cubic CNFs.

In this paper we give an alternative (and much simpler) proof of the same result (NP-hardness of clause and literal minimization for cubic pure Horn CNFs). Our polynomial time reduction is loosely based on the original reduction from [7], and uses the decomposition technique to prove the minimality of certain parts of the CNF constructed during the reduction.

The paper is structured as follows. Section 2 introduces basic Boolean terminology. Section 3 presents the developed method for proving CNF minimality. It consists of three subsections. The first subsection recalls results from [14] about exclusive sets of implicates of a Boolean function. The second subsection uses these results to formulate and prove the main theoretical contribution of the present paper, namely a general decomposition theorem that can be used to prove CNF minimality. The third subsection recalls further results from [14], this time about essential sets of implicates of a Boolean function. Section 4 introduces Horn and pure Horn CNFs and the forward chaining procedure which works on pure Horn CNFs together with few basic results related to this procedure. It also defines specific exclusive and essential sets of implicates of a pure Horn function which will be used later in the minimality proofs. Finally, Section 4 introduces all the decision problems studied in the rest of the paper (the NP-hard set cover problems used for the reductions and the clause and literal pure Horn minimization problems proved to be NP-hard). Section 5 contains the first simple example of the decomposition technique shown on a reduction very similar to the original proof from [7]. Section 6 deals with a second, more complicated, example of the decomposition technique shown on a reduction which is also loosely based on [7]. This final reduction proves the NP-hardness of clause and literal minimization for cubic pure Horn CNFs. The paper closes with a brief conclusions section.

2. Definitions

A *Boolean function* f on n propositional variables $x_1, \dots, x_n$ is a mapping $\{0,1\}^n \to \{0,1\}$. The propositional variables $x_1, \dots, x_n$ and their negations $\bar{x}_1, \dots, \bar{x}_n$ are called *literals* (*positive* and *negative literals*, respectively). An elementary disjunction of literals

$$C = \bigvee_{i \in I} \bar{x}_i \vee \bigvee_{j \in J} x_j \qquad (1)$$

is called a *clause*, if every propositional variable appears in it at most once, i.e. if $I \cap J = \emptyset$. The *degree* of a clause C is the number of literals in C. It is a well-known fact that every Boolean function f can be represented by a conjunction of clauses (see e.g. [19]). Such an expression is called a *conjunctive normal form* (or CNF) of the Boolean function f.

For two Boolean functions f and g we write $f \leqslant g$ if

$$\forall (x_1, \ldots, x_n) \in \{0,1\}^n\colon\ f(x_1, \ldots, x_n) = 1 \quad \Longrightarrow \quad g(x_1, \ldots, x_n) = 1. \tag{2}$$

Since each clause is in itself a Boolean function, formula (2) also defines the meaning of inequalities $C_1 \leqslant C_2$, $C_1 \leqslant f$, and $f \leqslant C_1$, where C_1, C_2 are clauses and f is a Boolean function.

We say that a clause C_1 *subsumes* another clause C_2 if $C_1 \leqslant C_2$ (e.g. the clause $\bar{x} \vee z$ subsumes the clause $\bar{x} \vee \bar{y} \vee z$). A clause C is called an *implicate* of a function f if $f \leqslant C$. An implicate C is called *prime* if there is no distinct implicate C' subsuming C, or in other words, an implicate of a function is prime if dropping any literal from it produces a clause which is not an implicate of that function.

It should be noted that a given Boolean function may have many CNF representations. We shall often identify a CNF with the set of clauses it contains and use these notions interchangeably. If two distinct CNFs, say ψ_1 and ψ_2, represent the same function, we say that they are *equivalent*, and denote this fact by $\psi_1 \equiv \psi_2$. CNF ψ representing function f is called *prime* if each clause of ψ is a prime implicate of function f. CNF ψ representing function f is called *irredundant* if dropping any clause from ψ produces a CNF that does not represent f. A clause containing two literals is called *quadratic* and a clause containing three literals is called *cubic*. A CNF in which every clause has degree at most three is called *cubic* and it will be denoted here as 3CNF. There are two common ways of measuring the "size" of a given CNF ψ:

- $|\psi|_c$ denotes the number of clauses in ψ,
- $|\psi|_\ell$ denotes the number of literals in ψ, i.e. $|\psi|_\ell = \sum_{C \in \psi} |C|$ where $|C|$ denotes the degree of C.

Furthermore, for a Boolean function f, let us denote by $\tau(f)$ the minimum number of clauses and by $\lambda(f)$ the minimum number of literals needed in a CNF representation of f, i.e. let

- $\tau(f) = \min\{|\psi|_c \mid \psi \text{ represents } f\}$,
- $\lambda(f) = \min\{|\psi|_\ell \mid \psi \text{ represents } f\}$.

Two clauses C_1 and C_2 are said to be *resolvable* if they contain exactly one complementary pair of literals, i.e. if there exists exactly one propositional variable that appears uncomplemented in one of the clauses and complemented in the other. That means that we can write $C_1 = \tilde{C}_1 \vee x$ and $C_2 = \tilde{C}_2 \vee \bar{x}$ for some propositional variable x and clauses $\tilde{C}_1$ and $\tilde{C}_2$ which contain no complementary pair of literals. The clauses C_1 and C_2 are called *parent clauses* and the disjunction $R(C_1, C_2) = \tilde{C}_1 \vee \tilde{C}_2$ is called the *resolvent* of the parent clauses C_1 and C_2. Note that the resolvent is a clause (does not contain a propositional variable and its negation). The following is an easy lemma [20].

Lemma 2.1. *Let C_1 and C_2 be two resolvable implicates of a Boolean function f. Then $R(C_1, C_2)$ is also an implicate of f.*

We say that a set of clauses $\mathcal{A}$ is closed under resolutions if given two resolvable clauses $C_1, C_2 \in \mathcal{A}$, their resolution $R(C_1, C_2)$ also belongs to $\mathcal{A}$. For an arbitrary set of clauses $\mathcal{C}$ the *resolution closure* of $\mathcal{C}$ denoted by $\mathcal{R}(\mathcal{C})$ is the smallest set of clauses with respect to inclusion, which contains clauses from $\mathcal{C}$ and which is closed under resolution. For a Boolean function f let us denote by $\mathcal{I}^p(f)$ the set of its prime implicates, and let $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}^p(f))$.

Example 2.2. Let us consider the following formula representing a function f:

$$\psi = (a \vee \bar{x} \vee z)(b \vee \bar{x} \vee z)(\bar{z} \vee y)(\bar{y} \vee x).$$

By performing resolutions on ψ one can obtain $\mathcal{I}^p(f)$:

$$\begin{aligned}\mathcal{I}^p(f) = \big\{&(a \vee \bar{x} \vee y), (a \vee \bar{x} \vee z), (a \vee \bar{y} \vee x), (a \vee \bar{y} \vee z), (a \vee \bar{z} \vee x), (a \vee \bar{z} \vee y),\\ &(b \vee \bar{x} \vee y), (b \vee \bar{x} \vee z), (b \vee \bar{y} \vee x), (b \vee \bar{y} \vee z), (b \vee \bar{z} \vee x), (b \vee \bar{z} \vee y),\\ &(\bar{z} \vee y), (\bar{y} \vee x), (\bar{z} \vee x)\big\}.\end{aligned}$$

By checking that there is no non-absorbed resolution it follows from the completeness of resolution that $\mathcal{I}^p(f)$ is indeed a complete list of prime implicates of f. Moreover by including also absorbed resolvents we obtain $\mathcal{I}(f)$:

$$\begin{aligned}\mathcal{I}(f) = \mathcal{I}^p(f) \cup \big\{&(a \vee b \vee \bar{x} \vee y), (a \vee b \vee \bar{x} \vee z), (a \vee b \vee \bar{y} \vee x),\\ &(a \vee b \vee \bar{y} \vee z), (a \vee b \vee \bar{z} \vee x), (a \vee b \vee \bar{z} \vee y)\big\}.\end{aligned}$$

3. A decomposition method for proving CNF minimality

In this section we shall first recall a definition and several key results from [14] concerning exclusive sets of implicates of a Boolean function which are needed for the decomposition method (Subsection 3.1). Then we present the decomposition theorem which is the main theoretical contribution of the present paper (Subsection 3.2). Finally, we recall a definition of essential sets of implicates and a key theorem about these sets from [14] (Subsection 3.3). This result is necessary for the application of the decomposition theorem in the two particular examples presented in Sections 5 and 6.

3.1. Exclusive sets of implicates of a Boolean function

We start by defining exclusive sets of implicates of a Boolean function and by stating some of their properties.

Definition 3.1. (See [14].) Let f be a Boolean function and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be a set of clauses. We shall say, that $\mathcal{X}$ is an *exclusive set of implicates of* f if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{I}(f)$ the following implication holds:

$$R(C_1, C_2) \in \mathcal{X} \quad \Longrightarrow \quad C_1 \in \mathcal{X} \text{ and } C_2 \in \mathcal{X},$$

i.e. the resolvent belongs to $\mathcal{X}$ only if both parent clauses are in $\mathcal{X}$. If function f is clear from the context, we shall simply say that $\mathcal{X}$ is an exclusive set.

We shall recall some of the properties of exclusive sets, which were proved in [14] and which we will use in this paper.

Lemma 3.2. *(See [14].) Let $\mathcal{A}, \mathcal{B} \subseteq \mathcal{I}(f)$ be exclusive sets of implicates of f, then both $\mathcal{A} \cup \mathcal{B}$ and $\mathcal{A} \cap \mathcal{B}$ are also exclusive subsets of f.*

Theorem 3.3. *(See [14].) Let f be an arbitrary Boolean function, let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{I}(f)$ be two distinct sets of clauses which both represent f, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of implicates of f. Then $\mathcal{C}_1 \cap \mathcal{X} \equiv \mathcal{C}_2 \cap \mathcal{X}$, i.e. both represent the same function.*

Based on this proposition we define an exclusive component of a Boolean function.

Definition 3.4. (See [14].) Let f be an arbitrary Boolean function, $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of implicates of f, and $\mathcal{C} \subseteq \mathcal{I}(f)$ be a set of clauses which represents f. The Boolean function $f_{\mathcal{X}}$ represented by the set $\mathcal{C} \cap \mathcal{X}$ is called the $\mathcal{X}$*-component* of the function f. We shall simply call a function g an *exclusive component of* f, if $g = f_{\mathcal{X}}$ for some exclusive subset $\mathcal{X} \subseteq \mathcal{I}(f)$.

Theorem 3.3 guarantees that the $\mathcal{X}$-component $f_{\mathcal{X}}$ is well defined for every exclusive set $\mathcal{X} \subseteq \mathcal{I}(f)$. Theorem 3.3 has the following corollary.

Corollary 3.5. *(See [14].) Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{I}(f)$ be two distinct sets of clauses such that $\mathcal{C}_1 \equiv \mathcal{C}_2 \equiv f$, i.e. such that both sets represent f, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Then $(\mathcal{C}_1 \setminus \mathcal{X}) \cup (\mathcal{C}_2 \cap \mathcal{X})$ also represents f.*

Example 3.6. Let us consider the formula ψ defined in Example 2.2 representing function f:

$$\psi = (a \vee \bar{x} \vee z)(b \vee \bar{x} \vee z)(\bar{z} \vee y)(\bar{y} \vee x).$$

We can find several exclusive sets of implicates of f, firstly $\mathcal{I}(f)$ itself forms an exclusive set. It can be shown (by checking all resolvable pairs of clauses) that the following two sets are also exclusive:

$$\mathcal{X}_a = \big\{(a \vee \bar{x} \vee y), (a \vee \bar{x} \vee z), (a \vee \bar{y} \vee x), (a \vee \bar{y} \vee z), (a \vee \bar{z} \vee x), (a \vee \bar{z} \vee y), (\bar{z} \vee y), (\bar{y} \vee x), (\bar{z} \vee x)\big\},$$

$$\mathcal{X}_b = \big\{(b \vee \bar{x} \vee y), (b \vee \bar{x} \vee z), (b \vee \bar{y} \vee x), (b \vee \bar{y} \vee z), (b \vee \bar{z} \vee x), (b \vee \bar{z} \vee y), (\bar{z} \vee y), (\bar{y} \vee x), (\bar{z} \vee x)\big\}.$$

It follows by Lemma 3.2 that the intersection

$$\mathcal{X}_a \cap \mathcal{X}_b = \big\{(\bar{z} \vee y), (\bar{y} \vee x), (\bar{z} \vee x)\big\}$$

forms an exclusive set, too. And finally it is not hard to check that $\{(\bar{x} \vee y)\}$ and $\{(\bar{y} \vee z)\}$ form two singleton exclusive sets. There are no other nonempty exclusive sets of implicates of f.

Note, that each of the above mentioned exclusive sets defines an exclusive component, e.g. clauses from $\mathcal{X}_a$ define an exclusive component of f, where $\psi \cap \mathcal{X}_a = (a \vee \bar{x} \vee z)(\bar{y} \vee x)(\bar{z} \vee y)$ is one of its representations.

3.2. *Decomposition properties of exclusive covers*

We shall start this subsection with a technical lemma and its corollary.

Lemma 3.7. *Let f be a Boolean function and let ψ be one of its CNF representations. Let $\mathcal{X}$ be an exclusive set of implicates of f, let $f_{\mathcal{X}}$ be the corresponding exclusive component, and let $\psi_{\mathcal{X}} = \psi \cap \mathcal{X}$ be its CNF representation inside of ψ. Let $\psi'_{\mathcal{X}}$ be an arbitrary CNF representation of $f_{\mathcal{X}}$, then $\psi' = (\psi \setminus \psi_{\mathcal{X}}) \cup \psi'_{\mathcal{X}}$ is again a CNF representation of f.*

Proof. In order to show that $\psi \equiv \psi'$ we have to show that any clause $C \in \psi$ is an implicate of ψ' and on the other hand that any clause $C' \in \psi'$ is an implicate of ψ. If C is a clause from ψ then there are two cases, either C belongs to $\psi_{\mathcal{X}}$, or not. If C does not belong to $\psi_{\mathcal{X}}$, then C is also in ψ' and thus it is an implicate of ψ'. If C belongs to $\psi_{\mathcal{X}}$, then C is an implicate of $f_{\mathcal{X}}$ and thus it is an implicate of a function represented by both $\psi_{\mathcal{X}}$ and $\psi'_{\mathcal{X}} = \psi' \cap \mathcal{X}$. The fact that C is an implicate of $\psi'_{\mathcal{X}}$ implies that C is an implicate of ψ'. We can argue in a symmetric way that a clause C' from ψ' is an implicate of ψ and hence $\psi \equiv \psi'$. $\square$

Corollary 3.8. *Let f be a Boolean function and let ψ be its minimum representation (with respect to either the number of clauses or literals). Let $\mathcal{X}$ be an arbitrary exclusive set of implicates of f, then $\psi_{\mathcal{X}} = \psi \cap \mathcal{X}$ is a minimum CNF representation of the exclusive $\mathcal{X}$-component (with respect to either the number of clauses or literals respectively).*

Proof. This follows directly from Lemma 3.7. Let us denote the $\mathcal{X}$-component of f by $f_{\mathcal{X}}$. If there would be a smaller CNF representation of $f_{\mathcal{X}}$ than $\psi_{\mathcal{X}}$, say $\psi'_{\mathcal{X}}$, we could plug it into ψ instead of $\psi_{\mathcal{X}}$ thus obtaining ψ' which would be a smaller CNF representation of f than ψ. That would be in contradiction with the assumption. $\square$

Note that the above two statements use only the fact that $\psi_{\mathcal{X}}$ and $\psi'_{\mathcal{X}}$ are logically equivalent and do not use the fact that $\mathcal{X}$ is an exclusive set.

Now we are ready to present a decomposition theorem which describes how an exclusive cover of a CNF, i.e. a collection of exclusive sets which covers all clauses of the given CNF, can be used to prove the minimality of the given CNF. This theorem is not specific to any particular subclass of CNFs and can be used for such proofs of minimality for arbitrary CNFs in all cases where an appropriate exclusive cover can be found. For brevity let us in the sequel denote $[r] = \{1, 2, \dots, r\}$ for positive integers $r \in \mathbb{Z}_+$.

Theorem 3.9. *Assume that ψ is a CNF representing a function g, and let the sets $\mathcal{X}_i$, $i = 1, \dots, p$ be an arbitrary collection of exclusive sets of implicates of g such that*

$$\psi \subseteq \hat{\mathcal{X}} = \bigcup_{i=1}^{p} \mathcal{X}_i.$$

For a nonempty subset $I \subseteq [p]$ let us introduce $\mathcal{X}_I = \bigcap_{i \in I} \mathcal{X}_i$ and let us denote by g_I the $\mathcal{X}_I$-component of g. Then, if for all nonempty subsets $I \subseteq [p]$ the CNF $\psi \cap \mathcal{X}_I$ is a minimum representation of g_I with respect to the number of clauses (literals), then ψ is also a minimum representation of g with respect to the number of clauses (literals).

Proof. Recall that by Lemma 3.2 both an intersection and union of exclusive sets is also an exclusive set. Thus for an arbitrary $I \subseteq [p]$ the set $\mathcal{X}_I$ is an exclusive subset of $\mathcal{I}(g)$ and hence g_I is well defined. We also get that $\hat{\mathcal{X}}$ is an exclusive set.

Let us first show this claim for the number of clauses. Consider an arbitrary prime CNF φ representing g with the minimum number of clauses. Then, for any exclusive set $\mathcal{X} \subseteq \mathcal{I}(g)$ and the corresponding $\mathcal{X}$-component $g_{\mathcal{X}}$ of g we must have $|\varphi \cap \mathcal{X}|_c = \tau(g_{\mathcal{X}})$ by Corollary 3.8. Thus, for all $I \subseteq [p]$ the equality

$$|\varphi \cap \mathcal{X}_I|_c = \tau(g_I) = |\psi \cap \mathcal{X}_I|_c \tag{3}$$

follows by our assumptions. Let us now consider the following chain of equalities:

$$|\varphi|_c = \left|\bigcup_{i=1}^{p}(\varphi \cap \mathcal{X}_i)\right|_c = \sum_{k=1}^{p} \sum_{\substack{I \subseteq [p] \\ |I|=k}} (-1)^{k-1} |\varphi \cap \mathcal{X}_I|_c = \sum_{k=1}^{p} \sum_{\substack{I \subseteq [p] \\ |I|=k}} (-1)^{k-1} |\psi \cap \mathcal{X}_I|_c = \left|\bigcup_{i=1}^{p}(\psi \cap \mathcal{X}_i)\right|_c = |\psi|_c.$$

The last equality follows by our assumption $\psi \subseteq \hat{\mathcal{X}}$, which also implies that the $\hat{\mathcal{X}}$-component of g is g itself, which then implies the first equality by the minimality of φ (possible clauses in φ outside of $\hat{\mathcal{X}}$ could be deleted as redundant contradicting the minimality of φ). The second and fourth equalities are by the sieve formula, while the third equality is implied by (3). To see the claim for the number of literals, it suffices to assume that φ represents g with the minimum number of literals, and to replace the $||_c$ norm by the $||_\ell$ norm everywhere in the above considerations. $\square$

The above theorem can be used for the proof of minimality of a given CNF in the following way. Find an appropriate covering of the input CNF by exclusive sets, where the key property of the cover is a sufficient simplicity of each individual exclusive set in the cover and of each nonempty intersection of these sets. Every exclusive set in question must have a simple enough structure so that the size of the minimum representation of the corresponding exclusive component can be determined and shown to be equal to the size of the intersection of the given CNF with the corresponding exclusive set. Of course, the particular way how to select a cover and how to prove the minimality of the intersections of the given CNF with the individual exclusive sets may depend on some specific properties of the studied CNF. In the last two sections of this paper we shall construct two particular examples of such minimality proofs. The specific properties of the CNFs appearing in these two examples will be stated in two simple lemmas in Subsection 4.1.

We shall conclude this subsection by a corollary of Theorem 3.9 that shows that the minimality assumption is really necessary only for $\psi \cap \mathcal{X}_i$, i.e. only one element sets $I = \{i\}$ have to be considered.

Corollary 3.10. *Assume that ψ is a CNF representing function g, and the sets $\mathcal{X}_i$, $i = 1, \ldots, p$ are exclusive subsets of $\mathcal{I}(g)$ such that*

$$\psi \subseteq \bigcup_{i=1}^{p} \mathcal{X}_i.$$

Let us denote by g_i the $\mathcal{X}_i$-component of g. Then, if for all $i \in [p]$ the CNF $\psi \cap \mathcal{X}_i$ is a minimum representation of g_i with respect to the number of clauses (literals), then ψ is also a minimum representation of g with respect to the number of clauses (literals).

Proof. The minimality of $\psi \cap \mathcal{X}_I$ for $|I| \geqslant 2$ follows by Lemma 3.2 (intersection of exclusive sets is exclusive) and by Corollary 3.8. Thus the assumptions of Theorem 3.9 are fulfilled. □

Let us note that a decomposition technique based on exclusive sets of implicates can be useful not only for minimality proofs but also for CNF manipulation that may in some cases lead to shorter logically equivalent CNFs. Given an input CNF (which is not minimum yet), consider an arbitrary exclusive component of the represented function. The corresponding part of the input CNF can be extracted, minimized independently of the rest of the input CNF, and then the minimal CNF (or any shorter CNF if a minimum cannot be achieved) of the exclusive component can be inserted back into the original CNF, thus getting a shorter CNF of the input function. This gives a heuristic minimization procedure which hinges on the ability to detect nontrivial exclusive sets of implicates. This procedure was described in [14]. A particular case of such iterative heuristic minimization procedure can be found in [13], although exclusive sets are used only implicitly there (the definition of exclusive sets and the results concerning them were presented in [14], much later than [13] was published). In some cases, the CNF manipulation based on the above idea can be shown to be not just a heuristic minimization procedure but really an optimization algorithm. An example of such a case can be found in [12].

Example 3.11. Let us consider the formula ψ from Example 2.2 representing function f:

$$\psi = (a \vee \bar{x} \vee z)(b \vee \bar{x} \vee z)(\bar{z} \vee y)(\bar{y} \vee x).$$

We want to show that ψ is a minimum CNF representation of f. We can see that $\psi \subseteq \mathcal{X}_a \cup \mathcal{X}_b$ and thus we can use Theorem 3.9 to argue that ψ is indeed a minimum CNF representation of f. To show this we have to check that $\psi \cap \mathcal{X}_a$, $\psi \cap \mathcal{X}_b$, and $\psi \cap \mathcal{X}_a \cap \mathcal{X}_b$ form minimum representations of the respective exclusive components. First let us consider $\psi \cap \mathcal{X}_a \cap \mathcal{X}_b = (\bar{x} \vee y)(\bar{y} \vee z)$. This is the only prime and irredundant representation of exclusive component defined by $\mathcal{X}_a \cap \mathcal{X}_b$ and thus it is also the only minimum one. Now, let us consider $\psi \cap \mathcal{X}_a = (a \vee \bar{x} \vee z)(\bar{z} \vee y)(\bar{y} \vee x)$. Note, that we already know that $(\bar{z} \vee y)(\bar{y} \vee x)$ is the only minimum representation of the exclusive component defined by $\mathcal{X}_a \cap \mathcal{X}_b$ and thus we cannot save anything in this part. There is only one more clause $(a \vee \bar{x} \vee z)$ and on the other hand we need to add at least one clause with variable a thus $\psi \cap \mathcal{X}_a$ is also a minimum representation of the respective exclusive component. The case of $\psi \cap \mathcal{X}_b$ is symmetrical. By Theorem 3.9 we thus get that ψ is indeed a minimum representation of f.

Let us observe that $\hat{\mathcal{X}} = \mathcal{X}_a \cup \mathcal{X}_b$ form a proper subset of $\mathcal{I}(f)$, it does not contain clauses containing both literals a and b, these clauses are redundant and do not appear in any prime and irredundant CNF representation of f. When searching for a minimum CNF representation of f it is indeed enough to consider clauses from $\hat{\mathcal{X}}$.

Let us moreover observe that it is important that the sets $\mathcal{X}_a$ and $\mathcal{X}_b$ we use to decompose ψ are exclusive sets of implicates of f. Without this exclusivity assumption Theorem 3.9 would not hold. To demonstrate it, let us consider another representation of function f by the following CNF ψ':

$$\psi' = (a \vee \bar{x} \vee y)(a \vee \bar{y} \vee z)(b \vee \bar{x} \vee y)(b \vee \bar{y} \vee z)(\bar{z} \vee y)(\bar{y} \vee x).$$

CNF ψ' is a prime and irredundant representation of f but it is not minimum because it contains six clauses where ψ contains only four. Now let us split ψ' into the following singleton sets $\mathcal{X}_1 = \{(a \vee \bar{x} \vee y)\}$, $\mathcal{X}_2 = \{(a \vee \bar{y} \vee z)\}$, $\mathcal{X}_3 = \{(b \vee \bar{x} \vee y)\}$, $\mathcal{X}_4 = \{(b \vee \bar{y} \vee z)\}$, $\mathcal{X}_5 = \{(\bar{z} \vee y)\}$, $\mathcal{X}_6 = \{(\bar{y} \vee x)\}$. It is true that $\psi' \subseteq \bigcup_{i=1}^{6} \mathcal{X}_i$ and moreover it is true that $\psi' \cap \mathcal{X}_i$ is a minimum representation of function g_i represented by $\mathcal{X}_i$, $i = 1, \ldots, 6$. Yet ψ' is not minimum representation of f and this is because the sets $\mathcal{X}_1, \ldots, \mathcal{X}_4$ are not exclusive sets of implicates of f and thus Theorem 3.9 does not apply in this case.

3.3. *Essential sets of implicates of a Boolean function*

When proving the minimality of certain representations of exclusive components the notion of an essential set (also originally defined in [14]) will become very useful.

Definition 3.12. (See [14].) Let f be a Boolean function and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be a set of clauses. We shall say, that $\mathcal{X}$ is an *essential set of clauses of* f if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{I}(f)$ the following implication holds:

$$R(C_1, C_2) \in \mathcal{X} \quad \Longrightarrow \quad C_1 \in \mathcal{X} \text{ or } C_2 \in \mathcal{X},$$

i.e. the resolvent belongs to $\mathcal{X}$ only if at least one of the parent clauses is in $\mathcal{X}$. If function f is clear from the context, we shall simply say that $\mathcal{X}$ is an essential set.

The key property of essential sets of implicates is stated in the following theorem.

Theorem 3.13. *(See [14].) Let $\mathcal{C} \subseteq \mathcal{I}(f)$ be an arbitrary set of clauses. Then $\mathcal{C}$ represents f if and only if $\mathcal{C} \cap \mathcal{E} \neq \emptyset$ for every nonempty essential set of clauses $\mathcal{E} \subseteq \mathcal{I}(f)$.*

Theorem 3.13 says that every CNF representation of function f must intersect every essential set of f. This implies an important fact for the lower bounds on the size of any CNF representation of f: if there exist k pairwise disjoint essential sets of f then any CNF representation of f contains at least k clauses.

Example 3.14. Let us for example consider the formula ψ representing the function f from Example 2.2. First any exclusive set of implicates of f is also an essential set of implicates. There are other essential sets of implicates, too, consider for example set $\mathcal{E}$ formed by implicates containing literal z, i.e.

$$\mathcal{E} = \{(a \vee \bar{x} \vee z), (a \vee \bar{y} \vee z), (b \vee \bar{x} \vee z), (b \vee \bar{y} \vee z), (a \vee b \vee \bar{x} \vee z), (a \vee b \vee \bar{y} \vee z)\}.$$

It is easy to see that $\mathcal{E}$ forms an essential set, because in order to get literal z in a resolvent it must come from one of the parent clauses. We shall consider this kind of essential sets in the next section.

4. Pure Horn functions and forward chaining procedure

A clause C defined by (1) is called *negative* if it contains no positive literals (i.e. if $J = \emptyset$). It is called *pure Horn* (or in some literature *definite Horn*) if it contains exactly one positive literal (i.e. if $|J| = 1$). To simplify notation, we shall sometimes write a pure Horn clause $C = \bigvee_{x \in S} \bar{x} \vee y$ simply as $C = S \vee y$. Each propositional variable $x \in S$ is called a *subgoal of* C and the propositional variable y is called the *head of* C. We shall denote $Head(C) = y$, $Subg(C) = S$, and $Vars(C) = S \cup \{y\}$.

A CNF is called *Horn* if it contains only negative and pure Horn clauses. A CNF is called *pure Horn* if it contains only pure Horn clauses. Finally, a Boolean function is called *Horn* if it has at least one representation by a Horn CNF, and similarly a Boolean function is called *pure Horn* if it has at least one representation by a pure Horn CNF.

It is known (see [21]) that each prime implicate of a Horn function is either negative or pure Horn, and each prime implicate of a pure Horn function is pure Horn. Thus, in particular, any prime CNF representing a Horn function is Horn, and any prime CNF representing a pure Horn function is pure Horn.

In verifying that a given clause is an implicate of a given pure Horn function, a very useful and simple procedure is the following. Let φ be a pure Horn CNF of a pure Horn function h. We shall define a *forward chaining* procedure which associates to any subset Q of the propositional variables of h a set $F_\varphi(Q)$ in the following way. The procedure takes as input the subset Q of propositional variables, initializes the set $F_\varphi(Q) = Q$, and at each step it looks for a pure Horn clause $S \vee y$ in φ such that $S \subseteq F_\varphi(Q)$, and $y \notin F_\varphi(Q)$. If such a clause is found, the propositional variable y is included into $F_\varphi(Q)$, and the search is repeated as many times as possible.

```
FORWARD CHAINING PROCEDURE($\mathcal{C}$, $Q$)
Input:            A set $\mathcal{C}$ of pure Horn clauses, and
                  a subset $Q$ of propositional variables.
Initialization:   Set $F_\mathcal{C}(Q) = Q$.
Main Step:        While $\exists C \in \mathcal{C}$: $Subg(C) \subseteq F_\mathcal{C}(Q)$ and $Head(C) \notin F_\mathcal{C}(Q)$
                  do $F_\mathcal{C}(Q) = F_\mathcal{C}(Q) \cup \{Head(C)\}$.
Output:           $F_\mathcal{C}(Q)$.
```

The following lemma, proved in [1], shows how the above procedure can help in determining whether a given clause is an implicate of a given CNF, or not.

Lemma 4.1. *Given a set $\mathcal{C}$ of pure Horn clauses, a subset Q of its propositional variables, and its variable $y \notin Q$, we have $y \in F_{\mathcal{C}}(Q)$ if and only if $Q \vee y$ is an implicate of the function represented by $\mathcal{C}$.*

In what follows we will frequently refer to CNFs as well as their sets of clauses, and thus if $\mathcal{C}$ is a set of clauses in CNF φ we shall write both $F_{\varphi}(Q) = F_{\mathcal{C}}(Q)$. Moreover, if φ' and φ'' are two distinct pure Horn CNF representations of a given pure Horn function h and if Q is an arbitrary subset of the propositional variables, then by Lemma 4.1 $F_{\varphi'}(Q) = F_{\varphi''}(Q)$ because φ' and φ'' have the same set of implicates. Therefore, the set of propositional variables reachable from Q by forward chaining depends only on the underlying function rather than on a particular CNF representation. For this reason, we shall also use the expression $F_h(Q)$ instead of $F_{\varphi}(Q)$ whenever we do not want to refer to a specific CNF. Furthermore, Lemma 4.1 has the following corollary.

Corollary 4.2. *Let φ' and φ'' be two CNFs on the same set V of variables. Then φ' and φ'' represent the same pure Horn function if and only if $F_{\varphi'}(X) = F_{\varphi''}(X)$ for every $X \subseteq V$.*

4.1. Exclusive and essential sets of pure Horn functions

In the two decomposition proofs that we present later in this paper the following type of exclusive sets of clauses will be used for the decomposition.

Lemma 4.3. *Let ψ be a prime, pure Horn CNF representing the function g over the set of variables $V = \{x_1, \dots, x_n\}$, let $W \subseteq V$ be such that $F_{\psi}(W) = W$, and define*

$$\mathcal{X}(W) = \{C \in \mathcal{I}(g) \mid vars(C) \subseteq W\}. \tag{4}$$

Then $\mathcal{X}(W)$ is an exclusive set of g.

Proof. Consider the binary assignment X defined as $x_j = 1$ for $j \in W$ and $x_j = 0$ otherwise. Since $F_{\psi}(W) = W$ holds, ψ does not have a clause which would have only its head outside of W, implying that $\psi(X) = g(X) = 1$.

Assume by contradiction now that $C_1, C_2 \in \mathcal{I}(g)$ are such that $R(C_1, C_2) \in \mathcal{X}(W)$, while $\{C_1, C_2\} \not\subseteq \mathcal{X}(W)$. Then by the definition of $\mathcal{X}(W)$ the clauses C_1, C_2 must be resolved over some variable $v \notin W$ while all other variables in C_1, C_2 must be in W (they compose the resolvent), and hence one of these implicates of g, say C_1, must have only its head outside of W, implying $g(X) \leqslant C_1(X) = 0$, which is a contradiction to $g(X) = 1$. □

While Lemma 4.3 will be used to decompose the set of implicates into small exclusive subsets, the following lemma will help in proving that the given representations of the individual exclusive components are minimal.

Lemma 4.4. *Let g be a pure Horn function, v be a variable of it, and define*

$$\mathcal{E}(v) = \{C \in \mathcal{I}(g) \mid v \text{ is the head of } C\}. \tag{5}$$

Then, $\mathcal{E}(v)$ is an essential set of g.

Proof. The head of the resolvent is the same as the head of one of the parent clauses, while the head of the other parent clause is the variable over which the parent clauses are resolved. So in fact in this case for every resolvent in $\mathcal{E}$ exactly one of the parent clauses is in $\mathcal{E}$. □

4.2. Horn minimization and set cover problems

Let us start by defining the decision problems, which we shall use later in this paper. We will use the decomposition method to prove NP-completeness of these problems. More specifically, we will use the decomposition method to prove the minimality of certain CNFs which are constructed in the polynomial transformations proving the NP-hardness of the below stated problems.

Horn minimization with respect to clauses (HMC)	
Instance:	A Horn CNF φ representing function h and integer $c \geqslant 0$.
Question:	Is $\tau(h) \leqslant c$?

Horn minimization with respect to literals (HML)	
Instance:	A Horn CNF φ representing function h and integer $\ell \geqslant 0$.
Question:	Is $\lambda(h) \leqslant \ell$?

Clause minimization of a pure Horn 3CNF (3HMC)	
Instance:	A pure Horn 3CNF φ representing function h and integer $c \geqslant 0$.
Question:	Is $\tau(h) \leqslant c$?

Literal minimization of a pure Horn 3CNF (3HML)	
Instance:	A pure Horn 3CNF φ representing function h and integer $\ell \geqslant 0$.
Question:	Is $\lambda(h) \leqslant \ell$?

All of the above defined decision problems clearly belong to the class NP. A certificate for a positive answer is in all four cases a Horn CNF of the appropriate length equivalent with the input CNF. Such a certificate is checkable in polynomial time because checking logical equivalence of two Horn CNFs amounts to testing whether each clause in one CNF is an implicate of the other CNF. Let us describe this in more details for the HMC problem. If we are given a Horn formula ψ we can check in polynomial time whether $|\psi|_c \leqslant c$ and whether ψ represents h. The latter step is done by checking if $\psi \equiv \varphi$. It is enough to check that every clause in ψ is an implicate of φ and on the other hand that every clause in φ is an implicate of ψ. This test is hard for general formulas, but for Horn CNFs this can be done in polynomial time The fact that a clause C is an implicate of a Horn CNF φ is equivalent to the fact that $\varphi \wedge \neg C$ is unsatisfiable (note, that if φ is Horn then also $\varphi \wedge \neg C$ is Horn as $\neg C$ is a conjunction of clauses of degree one). This unsatisfiability check can be done in linear time for Horn formula [22–24].

For the NP-hardness results we shall use a transformation from the following two NP-complete problems (which appear in [25] as two versions of the problem SP5).

Set Cover (SC)	
Instance:	An integer $k \geqslant 0$, a set of elements $U = \{u_1, \ldots, u_n\}$, a collection of subsets $\mathcal{S} = \{S_1, \ldots, S_m\} \subseteq \mathscr{P}(U)$ for which $\bigcup_{i=1}^m S_i = U$, where for every $S_i, S_j \in \mathcal{S}$ such that $i \neq j$ we have $S_i \nsubseteq S_j$, i.e. $\mathcal{S}$ has the Sperner property.
Question:	Is there a set $\mathcal{S}' \subseteq \mathcal{S}$ with $|\mathcal{S}'| \leqslant k$, for which $\bigcup_{S \in \mathcal{S}'} S = U$?

3-Set Cover (3SC)	
Instance:	An integer $k \geqslant 0$, a set of elements $U = \{u_1, \ldots, u_n\}$, a collection of subsets $\mathcal{S} = \{S_1, \ldots, S_m\} \subseteq \mathscr{P}(U)$ for which $\bigcup_{i=1}^m S_i = U$, where for every $S_i \in \mathcal{S}$ we have $|S_i| = 3$. We also assume that every set appears only once in $\mathcal{S}$, i.e. that $S_i \neq S_j$ for $i \neq j$.
Question:	Is there a set $\mathcal{S}' \subseteq \mathcal{S}$ with $|\mathcal{S}'| \leqslant k$, for which $\bigcup_{S \in \mathcal{S}'} S = U$?

Strictly speaking the cubic version of the problem SP5 in [25] is slightly different. It claims the NP-hardness for the variant of the set covering problem where every set has at most three elements rather than exactly three elements as required in (3SC). To see that this stricter condition does not influence the complexity status of the problem consider an instance $(U, \mathcal{S}, k)$ of the set cover problem such that $|S| \leqslant 3$ for all $S \in \mathcal{S}$. Then extend U with three new elements $U' = U \cup \{a, b, c\}$, extend every set of $\mathcal{S}$ of size 1 or 2 by adding elements a and/or b, as needed, and add a new set $H = \{a, b, c\}$ to $\mathcal{S}$. Then the new instance of set cover must have H in all covers (in order to cover c), and hence the minimum covers of the original instance and the new instance are in a one-to-one relation.

Furthermore let us denote by $\gamma(\mathcal{S})$ the number of sets in the minimum set cover for the family $\mathcal{S}$, i.e. let

$$\gamma(\mathcal{S}) = \min_{\mathcal{S}' \subseteq \mathcal{S}} \left\{ |\mathcal{S}'| \;\middle|\; \bigcup_{S \in \mathcal{S}'} S = U \right\}.$$

The problem (3SC) can be then reformulated as follows: given an instance $(U, \mathcal{S}, k)$ decide whether $\gamma(\mathcal{S}) \leqslant k$.

5. First example: the disjoint case

In this section we shall show, that problems HMC and HML are NP-hard. We shall use a slight modification of the construction used in [7] for showing the NP-hardness of the problem HMC. In [7] the authors used directed hypergraph terminology instead of Boolean terminology used in this paper, but the concepts are very similar. As in [7], our proof also uses the NP-hard problem SC for the transformation: it takes an instance of SC as an input and constructs a CNF as an instance of HMC and HML (the CNF is the same for both problems, the constructed numbers c for HMC and ℓ for HML are different).

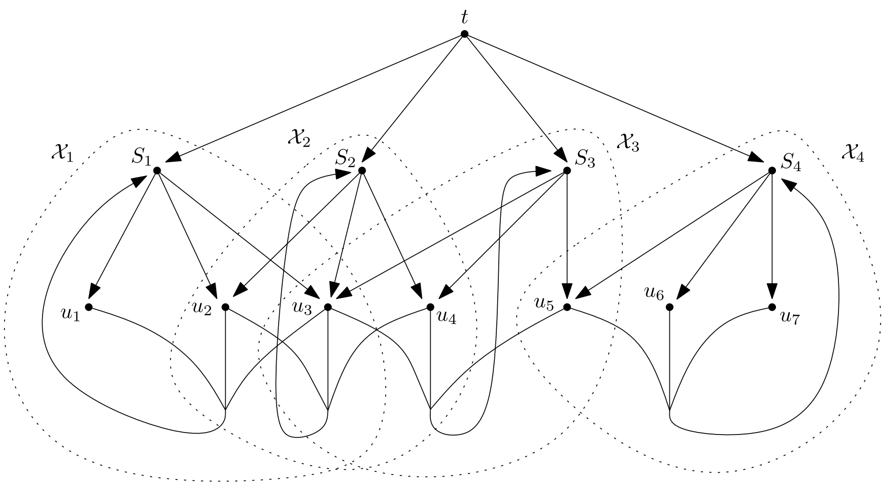

Fig. 1. An example of construction for sets $S_1 = \{u_1, u_2, u_3\}$, $S_2 = \{u_2, u_3, u_4\}$, $S_3 = \{u_3, u_4, u_5\}$, and $S_4 = \{u_5, u_6, u_7\}$. Dotted bubbles mark parts of the CNF corresponding to $\mathcal{X}_j$-components of g, $1 \leqslant j \leqslant 4$, considered in the proof of Theorem 5.4.

5.1. *Constructing the transformation from the set cover problem*

Let $(U, \mathcal{S}, k)$ be an instance of the SC problem, where $U = \{u_i \mid i \in [n]\}$, $\mathcal{S} = \{S_j \mid j \in [m]\}$, and for each $j \in [m]$ we have $S_j = \{u_{j(1)}, u_{j(2)}, \ldots, u_{j(\ell_j)}\}$, such that $j(1), j(2), \ldots, j(\ell_j)$ are distinct elements from $[n]$. Assume that elements of $U \cup \mathcal{S}$ denote also Boolean variables and let us introduce an additional Boolean variable t, different from the elements of U and $\mathcal{S}$. Finally, let us denote $V = U \cup \mathcal{S} \cup \{t\}$ to be the set of variables. Now let us consider the following sets of clauses on variables from V:

(i) $\alpha_j = (\bar{t} \vee S_j)$ for $j \in [m]$;
(ii) $\beta_{j,k} = (\bar{S}_j \vee u_{j(k)})$ for $j \in [m]$ and $k \in [\ell_j]$;
(iii) $\gamma_j = (\bar{u}_{j(1)} \vee \bar{u}_{j(2)} \vee \cdots \vee \bar{u}_{j(\ell_j)} \vee S_j)$ for $j \in [m]$.

Using the clauses of the second and third type let us define a CNF

$$\psi = \left(\bigwedge_{j \in [m]} \bigwedge_{k=1}^{\ell_j} \beta_{j,k} \right) \wedge \left(\bigwedge_{j \in [m]} \gamma_j \right),$$

and let g be the pure Horn function defined by ψ. Furthermore, using the clauses of the first type and the just specified function g let us define

$$\varphi_J = \left(\bigwedge_{j \in J} \alpha_j \right) \wedge \psi$$

for subsets $J \subseteq [m]$. Finally, let us denote by h the pure Horn function represented by $\varphi_{[m]}$ and by ℓ the sum of cardinalities of all sets in $\mathcal{S}$, i.e. $\ell = \sum_{j \in [m]} \ell_j$. You can see an example of the construction in Fig. 1. We shall show that the input set covering problem reduces to the minimization of the above defined pure Horn function h. In particular, we shall in the sequel arrive to the proof of the following statement.

Theorem 5.1. *Let $\mathcal{S}$ be the collection of subsets from the input instance of the SC problem and let h be the pure Horn function defined above. Then the following equivalences hold*

$$\gamma(\mathcal{S}) \leqslant k \quad \Longleftrightarrow \quad \tau(h) \leqslant c = k + \ell + m \quad \Longleftrightarrow \quad \lambda(h) \leqslant \ell = 2k + 3\ell + m.$$

Clearly, Theorem 5.1 shows exactly the desired result, i.e. that the problem SC reduces both to HMC and to HML. The proof of the theorem will consist of several steps. First, we shall prove that the function g constitutes an exclusive component of h and thus can be minimized independently. Then, we shall use the decomposition technique to show that CNF ψ is a minimum representation of g, i.e. that $\tau(g) = \ell + m$ and $\lambda(g) = 3\ell + m$. Furthermore, we shall show that CNF φ_J represents h if and only if the sets S_j for $j \in J$ form a cover of U. Finally, we shall prove that a minimum representation of h contains at most k quadratic clauses containing $\bar{t}$ if and only if there exists a set cover of U of size at most k. Note, that in this case we have $|\varphi_J|_c = |\psi|_c + k = \ell + m + k$, and $|\varphi_J|_\ell = |\psi|_\ell + 2k = 3\ell + m + 2k$ which are the bounds promised in Theorem 5.1.

5.2. *Decomposing the minimization of h*

Let us first observe that no clause in $\mathcal{I}(h)$ contains positive literal t, since none of the clauses of types (i)–(iii) contains it. Define W to be the set of all variables except variable t, and $\mathcal{X} \subseteq \mathcal{I}(h)$ to be the implicates of h not containing literal $\bar{t}$. Note that $W = F_h(W)$ and $\mathcal{X} = \mathcal{X}(W)$. Hence, Lemma 4.3 implies the following statement:

Lemma 5.2. *The set $\mathcal{X}$ is an exclusive subset of $\mathcal{I}(h)$.*

Obviously, $\varphi_{[m]} \cap \mathcal{X} = \psi$ and so ψ is a CNF representation of the $\mathcal{X}$-component of h by Theorem 3.3. Thus we get an easy corollary.

Corollary 5.3. *The function g is the $\mathcal{X}$-component of h. Consequently, given a CNF representation of h, the sub-CNF representing g can be replaced by an arbitrary minimum representation of g, and the resulting CNF still represents h.*

5.3. *Minimum representation of g*

We are going to argue in this subsection that ψ is a minimum representation of g with respect to both norms, i.e. that $\tau(g) = \ell + m$ and $\lambda(g) = 3\ell + m$.

Theorem 5.4. *The CNF ψ is a minimum (term and literal) representation of g, that is $\tau(g) = \ell + m$ and $\lambda(g) = 3\ell + m$.*

Proof. For every $j \in [m]$ let us define $W_j = F_\psi(\{S_j\})$ (the set of variables derivable by forwarding chaining on the CNF ψ from the single element set containing variable S_j), and set $\mathcal{X}_j = \mathcal{X}(W_j)$ as defined in Lemma 4.3 by Eq. (4). Then, by the statement of Lemma 4.3 the set $\mathcal{X}_j$ is an exclusive set of g for every $j \in [m]$. Since $\mathcal{S}$ has the Sperner property, we have that $W_j = \{S_j, u_{j(1)}, u_{j(2)}, \ldots, u_{j(\ell_j)}\}$ and $\psi \cap \mathcal{X}_j = \{\beta_{j,1}, \beta_{j,2}, \ldots, \beta_{j,\ell_j}, \gamma_j\}$ (note that $S_i \subset S_j$ would imply $S_i \in W_j$ and $\gamma_i \in \psi \cap \mathcal{X}_j$). Moreover no two clauses in $\psi \cap \mathcal{X}_j$ are resolvable which implies that $\psi \cap \mathcal{X}_j = \mathcal{X}_j$ is the unique prime representation of the $\mathcal{X}_j$-component of g, and hence it is obviously both clause minimum and literal minimum. It follows that the assumptions of Corollary 3.10 are met by the family of exclusive sets $\mathcal{X}_j$, $j \in [m]$. Now notice that there are ℓ clauses of type (ii) with a total of 2ℓ literals in ψ and m clauses of type (iii) with a total of $\ell + m$ literals in ψ, implying that $\tau(g) = \ell + m$ and $\lambda(g) = 3\ell + m$, which finishes the proof. $\square$

5.4. *Minimum representation of $\mathcal{I}(h) \setminus \mathcal{I}(g)$*

Let us define $\mathcal{D}$ to be the set of all prime implicates of f in $\mathcal{I}(h) \setminus \mathcal{X}$. Note that these are exactly all prime implicates of f containing variable t (i.e. containing literal $\bar{t}$). Let us at first determine which clauses belong to $\mathcal{D}$, certainly the clauses of type (i) belong to $\mathcal{D}$ as they belong to $\varphi_{[m]}$ and they do not belong to $\mathcal{X}$. We can easily show that the only other prime implicates which belong to $\mathcal{D}$ are of the following type:

(iv) $\alpha'_{j,k} = (\bar{t} \vee u_{j(k)})$ for $j \in [m]$, and $k \in [\ell_j]$.

Lemma 5.5. *All clauses of type* (iv) *belong to $\mathcal{I}(h)$.*

Proof. For every $j \in [m]$ and $k \in [\ell_j]$ we have $\alpha'_{j,k} = R(\alpha_j, \beta_{j,k})$. $\square$

Thus, by the definition of $\mathcal{D}$, all clauses of type (iv) belong to $\mathcal{D}$.

Lemma 5.6. *φ_J represents h if and only if $U \subseteq \bigcup_{j \in J} S_j$, that is iff $\{S_j \mid j \in J\}$ is a cover for $(U, \mathcal{S})$.*

Proof. We shall use forward chaining (in particular Corollary 4.2) to prove this statement. To this end, it suffices to show that $F_{\varphi_J}(W) = F_{\varphi_{[m]}}(W)$ for every subset $W \subseteq V$ of the variables if and only if $\{S_j \mid j \in J\}$ is a cover for $(U, \mathcal{S})$ (recall that $\varphi_{[m]}$ represents h by definition). We shall consider two cases:

1. $t \notin W$: In this case the forward chaining procedure works completely inside the exclusive component of h defined by g, where φ_J and $\varphi_{[m]}$ do not differ. Note that no clause of type (i), i.e. a clause from $\mathcal{D}$, may ever be used in the forward chaining procedure in this case. Thus $F_{\varphi_J}(W) = F_{\varphi_{[m]}}(W) = F_g(W) = F_h(W)$ follows immediately.
2. $t \in W$: Note that using all clauses of type (i) and of type (ii) we get $F_{\varphi_{[m]}}(\{t\}) = V$ and thus $F_{\varphi_{[m]}}(W) = V$ for every W containing t. Therefore, in order to prove our statement, it suffices to show that $F_{\varphi_J}(\{t\}) = V$ if and only if $\{S_j \mid j \in J\}$ is a cover for $(U, \mathcal{S})$. This fact can be verified by a step-by-step analysis of the forward chaining procedure on φ_J starting with set $\{t\}$. Namely, first we can observe that using clauses of type (i) the set $F_{\varphi_J}(\{t\})$ will include all variables S_j for

$j \in J$. Then, using clauses of type (ii), $F_{\varphi_J}(\{t\})$ will include all variables in $U' = \bigcup_{j \in J} S_j \subseteq U$, and subsequently, using clauses of type (iii), all variables S_j such that $S_j \subseteq U'$ (those which were not included before by clauses of type (i)). However, at this moment the forward chaining procedure stops, and no variable $u_j \in U \setminus U'$ will be included in $F_{\varphi_J}(\{t\})$. Hence $F_{\varphi_J}(\{t\}) = V$ if and only if $U \setminus U' = \emptyset$, i.e. if $\bigcup_{j \in J} S_j = U$. □

Lemma 5.7. *A minimum* (*both clause and literal*) *prime CNF representation of* h *involves exactly* $\gamma(\mathcal{S})$ *clauses from* $\mathcal{D}$.

Proof. Since no clause in $\mathcal{I}(h)$ involves literal t, and since the quadratic clauses in $\mathcal{D}$ involve all other variables of h (aside of t), $\mathcal{D}$ is exactly the set of prime implicates of h involving $\bar{t}$. Furthermore, in any CNF representation of h by Corollary 5.3 and by the proof of Lemma 5.5 (all clauses of types (iv) can be derived by resolution from clauses of type (i) and clauses from $\mathcal{I}(g)$), we can replace any clause of type (iv) by the corresponding clause of type (i) without changing the function the CNF represents, and without increasing its number of clauses and/or literals. Thus, it is enough to consider minimum representations of h of the type φ_J for subsets $J \subseteq [m]$, where g is replaced by a (clause or literal) minimum CNF representation of ψ. Then, our statements follow by Lemma 5.6. □

Now, using Lemma 5.7 we can conclude that $\tau(h) = \gamma(\mathcal{S}) + \tau(g)$ and $\lambda(h) = 2\gamma(\mathcal{S}) + \lambda(g)$ which put together with Theorem 5.4 finishes the proof of Theorem 5.1.

5.5. Notes on the original NP-hardness proof in [7]

In the original proof published in [7] the construction of $\varphi_{[m]}$ slightly differs.[1] The only difference is, that clauses of type (ii) are replaced by

(ii) $\gamma'_j = \bigvee_{i=1}^{n} \bar{u}_i \vee S_j$ for $j \in [m]$,

i.e. the set of subgoals of each such clause consists of all variables in U rather than only those which belong to the corresponding set S_j. Obviously, this changes the definition of the exclusive component g of h, and hence the arguments used in Subsection 5.3 cannot be used (the arguments used in Subsections 5.2 and 5.4 go through without any substantial change for the modified definition of $\varphi_{[m]}$). Unfortunately, the decomposition technique used in Subsection 5.3 does not work on the modified g as easily as in the previous case. However, the fact that ψ is again a minimum representation of g is still true, and can be proved (in a more complicated manner then in the previous case) as described below.

If we mimic the proof of Theorem 5.4 we get $W_j = \{S_j, u_{j(1)}, u_{j(2)}, \dots, u_{j(\ell_j)}\}$ and $\psi \cap \mathcal{X}_j = \{\beta_{j,1}, \beta_{j,2}, \dots, \beta_{j,\ell_j}\}$ (note that $\gamma'_j \notin \psi \cap \mathcal{X}_j$ in this case unless $S_j = U$ which makes the problem SC trivial). It is still true that no two clauses in $\psi \cap \mathcal{X}_j$ are resolvable and so again $\psi \cap \mathcal{X}_j = \mathcal{X}_j$ is the unique prime representation of the $\mathcal{X}_j$-component of g (which is both clause minimum and literal minimum), but this time the union of the $\mathcal{X}_j$ sets does not contain all clauses in ψ and so Corollary 3.10 cannot be used.

As argued above, every $\mathcal{X}_j$ is a unique prime representation of the $\mathcal{X}_j$-component of g and hence must be a part of a minimum length CNF representing h. What remains to be argued about is the minimality of the part of ψ outside of the union of all $\mathcal{X}_j$-components, i.e. of the conjunction of γ'_j for $j \in [m]$. For proving the minimality with respect to the number of clauses we can use Lemma 4.4 and Theorem 3.13. Clearly, given an arbitrary S_j, the clauses having S_j as a head form an essential set of g by Lemma 4.4, and each such essential set must be represented by at least one clause in any representation of g by Theorem 3.13. Since ψ intersects each such essential set in exactly one clause γ'_j, the minimality of ψ with respect to the number of clauses follows.

Let us comment in this context, that the original proof of Theorem 3 in [7] argues about the minimality of ψ by the following single sentence: "Note that if we take out from the hypergraph the node T and the hyperarcs leaving it the remaining hypergraph is nonredundant and no equivalent hypergraph with a smaller number of hyperarcs may exist." This is a true statement as argued above (in the previous paragraph), but in our opinion it is by no means a trivial observation (note that Lemma 4.4 and Theorem 3.13 were employed in our argument). To see the insufficiency of the argument, one can also try to replace the words "number of hyperarcs" with "sum of hyperarc sizes" in the above sentence from the proof in [7]. The new sentence seems to be just as plausible as the original one. Unfortunately, this time the claim is false which can be demonstrated as follows.

Let us construct an instance of the set cover problem such that ψ is not minimum with respect to the number of literals. Consider a set cover problem with $U = \{u_1, u_2, u_3\}$, $S_1 = \{u_1, u_2\}$, and $S_2 = \{u_2, u_3\}$ for which

$$\psi = (\bar{S}_1 \vee u_1) \wedge (\bar{S}_1 \vee u_2) \wedge (\bar{S}_2 \vee u_2) \wedge (\bar{S}_2 \vee u_3) \wedge (\bar{u}_1 \vee \bar{u}_2 \vee \bar{u}_3 \vee S_1) \wedge (\bar{u}_1 \vee \bar{u}_2 \vee \bar{u}_3 \vee S_2).$$

[1] The paper [7] uses a directed hypergraph terminology rather than pure Horn CNF terminology. Nevertheless, there is an obvious bijection between directed hypergraphs and pure Horn CNFs, so $\varphi_{[m]}$ can be thought of as a directed hypergraph, and conversely the directed hypergraph H constructed in Theorem 3 of [7] can be thought of as a pure Horn CNF.

It is possible to replace the last clause in ψ by $\bar{S}_1 \vee \bar{u}_3 \vee S_2$ which results in a logically equivalent CNF with a strictly smaller total number of literals.

6. Second example: the non-disjoint case

In this section we shall show, that problems 3HMC and 3HML are NP-hard. The proof is in many ways similar to the proof for HMC and HML in the previous section with two major differences. Firstly, the problem used for the transformation is 3SC instead of SC (note that the Sperner property of the system of subsets assumed for SC is satisfied automatically for 3SC), and secondly, the definition of the exclusive component g of h is different, which causes the exclusive sets used in the decomposition to intersect.

6.1. Constructing the transformation from the set cover problem

Let $(U, \mathcal{S}, k)$ be an instance of the 3SC problem, where $U = \{u_i \mid i \in [n]\}$, $\mathcal{S} = \{S_j \mid j \in [m]\}$, and for each $j \in [m]$ we have $S_j = \{u_{j(1)}, u_{j(2)}, u_{j(3)}\}$, such that $j(1), j(2), j(3)$ are distinct elements from $[n]$. Introduce further $Z = \{z_j \mid j \in [m]\}$, and assume that elements of $U \cup \mathcal{S} \cup Z$ denote also Boolean variables. Let as finally introduce an additional Boolean variable t, different from the elements of U, $\mathcal{S}$, and Z, and let us denote $V = U \cup \mathcal{S} \cup Z \cup \{t\}$ to be the set of variables. Now let us consider the following sets of clauses on variables from V:

(i) $\alpha_j = (\bar{t} \vee S_j)$ for $j \in [m]$;
(ii) $\beta_{j,k} = (\bar{S}_j \vee u_{j(k)})$ for $j \in [m]$, and $k = 1, 2, 3$;
(iii) $\gamma_j = (\bar{u}_{j(1)} \vee \bar{u}_{j(2)} \vee z_j)$, for $j \in [m]$; and
(iv) $\delta_j = (\bar{z}_j \vee \bar{u}_{j(3)} \vee S_j)$ for $j \in [m]$.

Using the clauses of the second, third and fourth type let us define a CNF

$$\psi = \left(\bigwedge_{j \in [m]} \bigwedge_{k=1}^{3} \beta_{j,k} \right) \wedge \left(\bigwedge_{j \in [m]} (\gamma_j \wedge \delta_j) \right),$$

and let g be the pure Horn function defined by ψ. Furthermore, using the clauses of the first type let us define

$$\varphi_J = \left(\bigwedge_{j \in J} \alpha_j \right) \wedge \psi$$

for subsets $J \subseteq [m]$. Finally, let us denote by h the pure Horn function represented by $\varphi_{[m]}$. We shall show that the input set covering problem reduces to the minimization of h. In particular, we shall in the sequel arrive to the proof of the following statement.

Theorem 6.1. *Let $\mathcal{S}$ be the collection of subsets from the input instance of the 3SC problem and let h be the pure Horn function defined above. Then the following equivalences hold*

$$\gamma(\mathcal{S}) \leqslant k \iff \tau(h) \leqslant c = k + 5m \iff \lambda(h) \leqslant \ell = 2k + 12m.$$

Clearly, Theorem 6.1 shows exactly the desired result, i.e. that the problem 3SC reduces both to 3HMC and to 3HML. The proof of the theorem will consist of several steps which are the same as in the disjoint case in the previous section. First, we shall note that the function g constitutes an exclusive component of h and thus can be minimized independently. Then, we shall prove that CNF ψ is a minimum representation of g, i.e. that $\tau(g) = 5m$ and $\lambda(g) = 12m$ (this step is different from the disjoint case). We shall conclude the proof by showing, that (just as in the disjoint case) CNF φ_J represents h if and only if the sets S_j for $j \in J$ form a cover of U, which in turn implies that a minimum representation of h contains at most k quadratic containing $\bar{t}$ if and only if there exists a set cover of U of size at most k.

6.2. Decomposing the minimization of h

This subsection is a carbon copy of Subsection 5.2 so let us just state the results again.

Lemma 6.2. *The set $\mathcal{X} \subseteq \mathcal{I}(h)$ of implicates of h not involving variable t is an exclusive subset of $\mathcal{I}(h)$.*

Corollary 6.3. *The function g is the $\mathcal{X}$-component of h. Consequently, given a CNF representation of h, the sub-CNF representing g can be replaced by an arbitrary minimum representation of g, and the resulting CNF still represents h.*

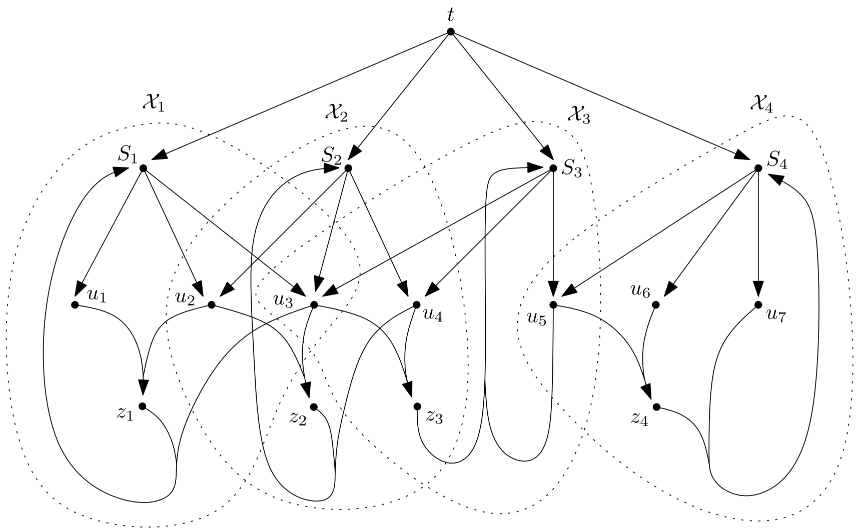

Fig. 2. An example of construction for sets $S_1 = \{u_1, u_2, u_3\}$, $S_2 = \{u_2, u_3, u_4\}$, $S_3 = \{u_3, u_4, u_5\}$, and $S_4 = \{u_5, u_6, u_7\}$. Dotted bubbles mark parts of the CNF corresponding to $\mathcal{X}_j$-components of g, $1 \leqslant j \leqslant 4$, considered in the proof of Theorem 6.4.

6.3. *Minimum representation of g*

We are going to argue in this subsection that ψ is a minimum representation of g with respect to both norms, i.e. that $\tau(g) = 5m$ and $\lambda(g) = 12m$.

Theorem 6.4. *The CNF ψ is a minimum (clause and literal) representation of g, that is $\tau(g) = 5m$ and $\lambda(g) = 12m$.*

Proof. For every $j \in [m]$ let us define $W_j = F_\psi(\{S_j\})$ (the set of variables derivable by forwarding chaining on the CNF ψ from the single element set containing variable S_j), and set $\mathcal{X}_j = \mathcal{X}(W_j)$ as defined in Lemma 4.3 by Eq. (4). Then, by the statement of Lemma 4.3 the set $\mathcal{X}_j$ is an exclusive set of g for every $j \in [m]$. We shall show that the assumptions of Corollary 3.10 are met by the family of exclusive sets $\mathcal{X}_j$, $j \in [m]$, which will finish the proof. To arrive to this end, let us set $I_j = \{i \in [m] \mid \{i(1), i(2)\} \subseteq \{j(1), j(2), j(3)\}\}$ (in Fig. 2 we have $I_1 = \{1, 2\}$, $I_2 = \{2, 3\}$, $I_3 = \{3\}$, and $I_4 = \{4\}$), and observe the following:

(a) $W_j = \{S_j, u_{j(1)}, u_{j(2)}, u_{j(3)}\} \cup \{z_i \mid i \in I_j\}$. In fact the only clauses of ψ playing a role in the forward chaining derivation of W_j are $\beta_{j,k}$ for $k = 1, 2, 3$ and γ_i for indices $i \in I_j$. Note that for any such $i \neq j$, the index $i(3)$ must be outside of $\{j(1), j(2), j(3)\}$, and hence the forward chaining can never use any z_i for $i \neq j$ as a subgoal to further enlarge W_j.
(b) $\psi \cap \mathcal{X}_j = \{\beta_{j,1}, \beta_{j,2}, \beta_{j,3}, \delta_j\} \cup \{\gamma_i \mid i \in I_j\}$.

In order to use Corollary 3.10 we have to verify that $\psi \cap \mathcal{X}_j$ is a minimum representation of g_j, the $\mathcal{X}_j$-component of g.

For proving the minimality with respect to the number of clauses we can (similarly as in Subsection 5.5) use Lemma 4.4 and Theorem 3.13. Clearly, given an arbitrary variable $v \in W_j$, the clauses having v as a head form an essential set $\mathcal{E}(v)$ of g_j by Lemma 4.4, and each such essential set must be represented by at least one clause in any representation of g_j by Theorem 3.13. Since $\psi \cap \mathcal{X}_j$ intersects each such essential set $\mathcal{E}(v)$ in exactly one clause, the minimality of $\psi \cap \mathcal{X}_j$ with respect to the number of clauses follows.

Showing that $\psi \cap \mathcal{X}_j$ is also a minimum representation of g_j with respect to the number of literals needs a bit more work. As we have observed in the previous paragraph, any CNF representation of g_j must intersect every $\mathcal{E}(v)$, $v \in W_j$, so the only hope to make the CNF shorter would be to replace the unique clause from $\mathcal{E}(v)$ in $\psi \cap \mathcal{X}_j$ by a shorter clause from the same $\mathcal{E}(v)$. However, since there are no linear implicates of g_j, all quadratic implicates are already the shortest possible. Hence the only possible shortening could occur by replacing a cubic clause of type (iii) or (iv) by a quadratic clause with the same head. However, as one can easily check by subsequently performing forward chaining on $\psi \cap \mathcal{X}_j$ starting with set $\{v\}$ for all $v \in W_j$, the only quadratic clauses in $\mathcal{I}(g_j)$ not explicitly present in $\psi \cap \mathcal{X}_j$ are clauses $\bar{S}_j \vee z_i$, $i \in I_j$. So the only possible replacement of a cubic clause by a quadratic clause in the same $\mathcal{E}(v)$ is to replace γ_i for some $i \in I_j$ by the corresponding $\bar{S}_j \vee z_i$. Now one can check that no such replacement (or set of such replacements) maintains the logical equivalence of the two CNFs as forward chaining starting with $\{u_{j(1)}, u_{j(2)}\}$ no longer derives z_i after the replacement. Hence $\psi \cap \mathcal{X}_j$ is a literal minimum representation of g_j. $\square$

6.4. *Minimum representation of* $\mathcal{I}(h) \setminus \mathcal{I}(g)$

This subsection is very similar to Subsection 5.4, so we shall point out only the differences. Let us again consider the set $\mathcal{D}$ of all prime implicates in $\mathcal{I}(h) \setminus \mathcal{X}$, this is the set of all prime implicates containing literal $\bar{t}$. It is immediate that the clauses of type (i) all belong to $\mathcal{D}$. The only other implicates belonging to $\mathcal{D}$ have one of the following types:

(v) $\alpha'_{j,k} = (\bar{t} \vee u_{j(k)})$ for $j \in [m]$, and $k = 1, 2, 3$; and
(vi) $\alpha''_j = (\bar{t} \vee z_j)$, for $j \in [m]$.

Lemma 6.5. *All clauses of types* (v) *and* (vi) *belong to* $\mathcal{I}(h)$.

Proof. For every $j \in [m]$ and $k = 1, 2, 3$ we have that $\alpha'_{j,k} = R(\alpha_j, \beta_{j,k})$ and $\alpha''_j = R(\alpha'_{j,1}, R(\alpha'_{j,2}, \gamma_j))$. □

It follows that $\mathcal{D}$ is formed by clauses of types (i), (v), and (vi).

Lemma 6.6. φ_J *represents h if and only if* $U \subseteq \bigcup_{j \in J} S_j$, *that is iff* $\{S_j \mid j \in J\}$ *is a cover for* $(U, \mathcal{S})$.

Proof. The proof is almost the same as the proof of Lemma 5.6. The only difference is that clauses of types (ii) and (iii) have to be used together where only type (ii) was used in Lemma 5.6, and of course type (iv) has to be used where type (iii) was used before. □

Lemma 6.7. *A minimum* (*both clause and literal*) *prime representation of h involves exactly* $\gamma(\mathcal{S})$ *clauses from* $\mathcal{D}$.

Proof. Again, the proof is almost the same as the proof of Lemma 5.7. The only difference is that clauses of types (v) and (vi) play the role played by type (iv) in Lemma 5.7. □

Now, using Lemma 6.7 we can conclude that $\tau(h) = \gamma(\mathcal{S}) + \tau(g)$ and $\lambda(h) = 2\gamma(\mathcal{S}) + \lambda(g)$ which put together with Theorem 6.4 finishes the proof of Theorem 6.1.

7. Conclusions

In this paper we have designed a procedure for proving the minimality of a given CNF, i.e. for proving that no shorter logically equivalent CNF exists, where the length of a CNF is either the number of clauses or the total number of literal occurrences in the given CNF. This procedure rests on a decomposition of the input CNF into so-called exclusive components where the minimality of each component can be then proved independently. The minimality of all components suffices for the minimality of the input CNF. How to decompose the input CNF (and whether a nontrivial decomposition is possible at all) as well as how to prove the minimality of the individual components depends on the properties of the input CNF. We have shown two examples of such decompositions and component minimality proofs. These examples give alternative proofs to known minimization complexity results, namely NP-hardness of minimization for Horn CNFs and NP-hardness of minimization for Horn 3CNFs, in both cases with respect to both the number of clauses and the total number of literal occurrences.

Acknowledgements

The first author also thanks for partial support the NSF (Grants CMMI-0856663 and IIS-1161476). The second and third author gratefully acknowledge a support by the Czech Science Foundation (grant P202/10/1188).

References

[1] P. Hammer, A. Kogan, Optimal compression of propositional horn knowledge bases: Complexity and approximation, Artif. Intell. 64 (1993) 131–145.
[2] P.L. Hammer, A. Kogan, Knowledge compression – logic minimization for expert systems, in: Proceedings of IISF/ACM Japan International Symposium, World Scientific, Singapore, Tokyo, 1994, pp. 306–312.
[3] S.A. Cook, The complexity of theorem-proving procedures, in: Proceedings of the Third Annual ACM Symposium on Theory of Computing, STOC '71, ACM, New York, NY, USA, 1971, pp. 151–158.
[4] C. Umans, The minimum equivalent DNF problem and shortest implicants, J. Comput. Syst. Sci. 63 (2001) 597–611.
[5] C. Umans, T. Villa, A.L. Sangiovanni-Vincentelli, Complexity of two-level logic minimization, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 25 (2006) 1230–1246.
[6] D. Buchfuhrer, C. Umans, The complexity of Boolean formula minimization, J. Comput. Syst. Sci. 77 (2011) 142–153.
[7] G. Ausiello, A. D'Atri, D. Sacca, Minimal representation of directed hypergraphs, SIAM J. Comput. 15 (1986) 418–431.
[8] E. Boros, O. Čepek, On the complexity of Horn minimization, Technical Report 1-94, RUTCOR Research Report RRR, Rutgers University, New Brunswick, NJ, 1994.
[9] O. Čepek, Structural properties and minimization of Horn Boolean functions, Ph.D. dissertation, Rutgers University, New Brunswick, NJ, October 1995.

[10] D. Maier, Minimal covers in the relational database model, J. ACM 27 (1980) 664–674.
[11] P. Hammer, A. Kogan, Quasi-acyclic propositional Horn knowledge bases: Optimal compression, IEEE Trans. Knowl. Data Eng. 7 (1995) 751–762.
[12] E. Boros, O. Čepek, A. Kogan, P. Kučera, A subclass of Horn CNFs optimally compressible in polynomial time, Ann. Math. Artif. Intell. 57 (2009) 249–291.
[13] E. Boros, O. Čepek, A. Kogan, Horn minimization by iterative decomposition, Ann. Math. Artif. Intell. 23 (1998) 321–343.
[14] E. Boros, O. Čepek, A. Kogan, P. Kučera, Exclusive and essential sets of implicates of Boolean functions, Discrete Appl. Math. 158 (2010) 81–96.
[15] Š. Bigoš, Hornovské formule, Master's thesis, Charles University in Prague, Faculty of Mathematics and Physics, 2010 (in Czech).
[16] O. Čepek, P. Kučera, On the complexity of minimizing the number of literals in Horn formulae, Technical Report 11, RUTCOR Research Report RRR, Rutgers University, New Brunswick, NJ, 2008.
[17] A. Bhattacharya, B. DasGupta, D. Mubayi, G. Turán, On approximate Horn formula minimization, in: S. Abramsky, C. Gavoille, C. Kirchner, F. Meyer auf der Heide, P. Spirakis (Eds.), Lecture Notes in Computer Science, vol. 6198, Springer, Berlin/Heidelberg, 2010, pp. 438–450.
[18] E. Boros, A. Gruber, Hardness results for approximate pure Horn CNF formulae minimization, in: Proceedings of International Symposium on AI and Mathematics (ISAIM), 2012.
[19] M. Genesereth, N. Nilsson, Logical Foundations of Artificial Intelligence, Morgan Kaufmann, Los Altos, CA, 1987.
[20] H.K. Büning, T. Lettmann, Propositional Logic: Deduction and Algorithms, Cambridge University Press, New York, NY, USA, 1999.
[21] P. Hammer, A. Kogan, Horn functions and their DNFs, Inf. Process. Lett. 44 (1992) 23–29.
[22] W. Dowling, J. Gallier, Linear time algorithms for testing the satisfiability of propositional Horn formulae, J. Log. Program. 3 (1984) 267–284.
[23] A. Itai, J. Makowsky, Unification as a complexity measure for logic programming, J. Log. Program. 4 (1987) 105–117.
[24] M. Minoux, LTUR: A simplified linear time unit resolution algorithm for Horn formulae and computer implementation, Inf. Process. Lett. 29 (1988) 1–12.
[25] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, San Francisco, 1979.