

SEMINARSKI RAD IZ AUTOMATSKOG REZONOVANJA

-Implementacija DP procedure za iskaznu logiku-

Davis-Putnam procedura je metod za rešavanje problema zadovoljivosti KNF formula (SAT problema). DP procedura je zasnovana na pravilu rezolucije.

Koraci izvršavanja algoritma:

1. brišemo tautologične klaze (inače imamo beskonačnu rekurziju),
2. dokle god nije F prazna ili F ne sadrži praznu klauzu,
 - 2.1. propagacija jediničnih klauza (engl. unit propagation),
 - 2.2. eliminacija "čistih" literala (engl. pure literal) I
 - 2.3. eliminacija promenljive (engl. variable elimination).

Propagacija jediničnih klauza

Propagacija jediničnih klauza zasniva na pravilima $1 \text{ or } A = 1$ i $0 \text{ or } A = A$.

Kada se u formuli nalazi klauza koja sadrži samo jedan literal, naziva se jedinična klauza.

Propagacija jedinične klauze podrazumeva određivanje takvih klauza I propagiranje literala kroz ostale klauze.

Procedura se realizuje:

- Odredimo klauze koje imaju samo jedan literal L – kako nemaju ni jedan drugi literal neophodno je da je on tačan,
- propagiramo kroz ostale klauze – tamo gde se pojavljuje literal L tu klauzu možemo da uklonimo, tamo gde se pojavljuje suprotnog polariteta uklanjamo samo literal iz klauze.

Eliminacija "čistih" literala

Kada se u formuli neki literal javlja samo u svojoj pozitivnoj ili negativnoj formi (ne i u suprotnoj formi) taj literal se naziva čisti literal. Eliminacija čistih literala se odnosi na proces uklanjanja svih klauza koje sadrže te čiste litrale iz formule.

Eliminacija promenljive

Kada nije moguće primeniti ni propagaciju jedinične klauze, ni eliminaciju čistih literala neophodno je odrediti jedan literal po kome ćemo vršiti rezoluciju.

Rezolucija se vrši kroz sledeći niz koraka:

- Odaberemo literal L,
- Odredimo sve klauze koje sadrže L – allClauseContainL ,
- Odredimo sve klauze koje sadrže $\sim L$ – $\text{allClauseContainNotL}$,
- Vršimo rezoluciju za svake dve klauze (c_1, c_2), gde $c_1 \in \text{allClauseContainL}$, $c_2 \in \text{allClauseContainNotL}$,
- Ubacimo u skup svih klauza klauze koje su nastale kao rezultat rezolucije,

- Izbrišemo iz skupa svih klauza klauze koje se nalaze u skupovima `allClauseContainL` i `allClauseContainNotL`.

Implementacija

CNF (NormalForm) je implementirana kao lista klauza. Klauzu implementiramo kao skup. Funkcija koja sprovodi implementaciju DP procedure je `bool davis_putnam (NormalForm& cnf)`. Ulaz je formula iskazne logike u KNF, izlaz je `true` (SAT) ili `false` (UNSAT).

Na početku algoritma za svaku klauzu pozovemo funkciju `remove_tautology_clause`, koja briše klauzu ako je tautologična i vraća iterator na sledeću klauzu koju treba obraditi.

U beskonačnoj petlji izvršavamo:

- 1) *Propagaciju jedinične klauze* realizujemo tako što nalazimo klauze veličine 1, neka je literal sadržan u toj klauzi L . Funkcijom `unit_propagation` iz `cnf` brišemo sve klauze koje sadrže L , i uklanjamo iz svih klauza literal $\sim L$.
- 2) *Eliminaciju čistih literala* realizujemo tako što funkcijom `findPureLiterals` nađemo sve literalne u jednom polaritetu i smeštamo ih u skup `allLiteral`. Nakon toga uklanjamo sve klauze koje sadrže neki od literala iz `allLiteral`.
- 3) *Proveravamo uslove za izlaz iz petlje*: ako smo izveli praznu klauzu vratimo `false`, ako smo izbrisali sve klauze iz `cnf` vratimo `true`.
- 4) Na kraju iteracije petlje *eliminiramo promenljivu*. Uzimamo prvu u prvoj klauzi, odredimo dva skupa klauza koje sadrže promenljivu u pozitivnom i negativnom obliku (`tmp1`, `tmp2`), izvršimo `resolveClauses` za svake dve klauze uz `tmp1` i `tmp2` redom, izbrisemo iz `cnf` sve elemente `tmp1` i `tmp2`. Brisanje jedne klauze je vremenske složenosti $O(1)$, jer smo koristili funkciju `erase` za iterator i `cnf` je implementiran preko liste.

Ostaje još da pojasnimo pomoćnu funkciju `resolveClauses`. Ona je implementirana tako da prima `cnf`, `c1`, `c2` i L . Pretpostavljamo da je `c1` klauza koja sadrži L , `c2` klauza koja sadrži $\sim L$. Prođemo sve literalne iz `c1` i ubacimo u novi skup sve osim L , prođemo kroz sve literalne iz `c2` ubacimo sve literalne osim $\sim L$. Na kraju samo proverimo da li je klauza tautologija, ako nije ubacimo je u `cnf`.

Pokretanje programa

Program se prevodi komandom:

```
$g++ -std=c++17 -o izvrsni main.cpp
```

Za izvršavanje programa potreban je fajl u kom je smeštena formula u DIMACS formatu (npr `in.txt`). Program se izvršava komandom:

`$/izvrsni in.txt`

Rezultat se ispisuje na standardni ulaz. Moguće je preusmeriti izlaz u fajl komandom:

`$/izvrsni in.txt > out.txt`

Više o DIMACS formatu

DIMACS CNF predstavlja standardni format zapisa KNF formula u tekstualne datoteke. Koristi se kao ulaz SAT rešavača.

Osnovne karakteristike DIMACS formata:

- **Komentari:** Linije koje počinju sa znakom 'c' koriste se za komentare i ne utiču na samu formulu. Ovi komentari se često koriste za opisivanje problema ili dodatne informacije.
- **Specifikacija:** Linija koja počinje sa znakom 'p' se koristi za specificiranje problema. Nakon 'p' se navodi informacija o broju promenljivih, broju klauza i drugim relevantnim informacijama. Na primer, "p cnf 100 450" označava da je problem u konjunktivnoj normalnoj formi (cnf) sa 100 promenljivih i 450 klauza.
- **Klauze:** Linije koje predstavljaju klauze sadrže literale odvojene razmakom ili tabulatorom, a klauze se završavaju nulom (0) koja označava kraj klauze. Svaki literal je broj koji predstavlja promenljivu, a negativan broj označava negaciju te promenljive. Na primer, klauza $(x_1 \vee \neg x_2 \vee x_3)$ bi se predstavila kao "1 -2 3 0" u DIMACS formatu.
- **Numeracija promenljivih:** Promenljive se numerišu pozitivnim celim brojevima, pri čemu 1 predstavlja prvu promenljivu, 2 drugu promenljivu, itd. Negacije promenljivih se označavaju negativnim brojevima.