Problemi planiranja i svođenje na SAT

Milan Mitreski

Univerzitet u Beogradu, Matematički fakultet

4. septembar 2025.

Sažetak

Problem planiranja predstavlja jednu od prvih primena SAT rešavača pri rešavanju opštih problema pretrage. U ovom radu opisujemo postupak rešavanja problema planiranja korišćenjem SAT rešavača uz prateću programsku implementaciju koju primenjujemo na rešavanje jednog specifičnog problema koji se može svesti na problem planiranja.

Sadržaj

1 Uvod

1.1 Problem planiranja. Klasično planiranje

Planiranje je grana veštačke inteligencije koja se bavi procesom pronalaženja *plana* odnosno niza *akcija* koji dovodi do postizanja određenog *cilja*. *Problem planiranja* najčešće je definisan *početnim stanjem*, *skupom akcija* i *ciljnim stanjem* odnosno *ciljem*. U svojoj najosnovnijoj formi, zvanoj *klasično planiranje*, početno stanje je *jedinstveno* i akcije su *determinističke*.

Pristupi problemima planiranja su između 60-ih i 90-ih godina prethodnog veka podrazumevali *specijalizovanu pretragu* odnosno pretragu u okviru domena specifičnog problema. Radovi Kautz[1]-a i Selman[2]-a iz 1992. i 1996. su dali drugačiji pristup rešavanju problema planiranja svođenjem na opštiji problem ispitivanja zadovoljivosti iskazne formule tj. na SAT problem. Ovakav pristup je kasnije primenjen i na druge probleme, kao što su računarska verifikacija modela i validacija.

Nažalost, u opštem slučaju (uz pretpostavku da P≠PSPACE) ne postoji redukcija problema klasičnog planiranja na SAT problem u polinomijalnom vremenu. Iskazna formula koja predstavlja problem planiranja može, u najgorem slučaju, biti eksponencijalne složenosti u odnosu na veličinu početnog problema. Međutim, veličina ove formule zavisi od *dužine plana* odnosno potrebnog broja akcija da se iz početnog stanja dođe do cilja. Veliki broj problema planiranja rešiv je planovima čija je dužina polinomijalne složenosti, što ukazuje na činjenicu da će veličina odgovarajuće iskazne formule takođe biti polinomijalne složenosti. Za ovakve probleme je u praksi moguće konstruisati traženu iskaznu formulu (u polinomijalnom vremenu) kojom taj problem svodimo na SAT.

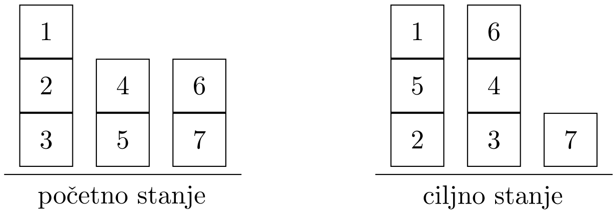
Upotreba SAT rešavača u problemima planiranja podrazumeva konstrukciju iskazne formule za problem postojanja plana ograničene dužine tj. ispitivanja da li postoji plan određene dužine n koji rešava problem planiranja. Za dato n, dobijena iskazna formula je zadovoljiva ako i samo ako postoji plan dužine n koji rešava problem planiranja.

Izazovi sa kojim se suočavamo u ovakvom pristupu su efikasna konstrukcija iskazne formule $\mathcal{F}_n$ za datu dužinu plana n, kao i dobar izbor mogućih vrednosti n tako da što brže dođemo do zadovoljive iskazne formula $\mathcal{F}_n$.

1.2 Blocks World (BW) problem planiranja

Prikazaćemo jedan ilustrativan primer svođenja problema planiranja na SAT problem – u pitanju je *Blocks World (BW) planning* problem [BW]. BW se sastoji od *blokova* koji su naslagani u *kule* na *stolu*, pri čemu je potrebno da početno stanja blokova transformišemo u ciljno stanje, a jedini dozvoljeni potez je uklanjanje jednog bloka sa vrha kule i njegovo postavljanje na vrh druge kule ili započinjanje nove kule odnosno postavljanje bloka na sto. BW problem planiranja jeste problem pronalaska optimalne transformacije (u smislu broja poteza) početnog stanja u ciljno stanje. Napomenimo da je BW jedan NP-težak problem. Na Slici 1 prikazana je jedna instanca problema BW kao i odgovarajući optimalan plan.

[1]Henry Kautz – američki informatičar
[2]Bart Selman – američki informatičar



1. postavi 1 na sto	2. postavi 2 na sto
3. postavi 4 na 3	4. postavi 5 na 2
5. postavi 1 na 5	6. postavi 6 na 4

Slika 1: Instanca BW problema planiranja i optimalni plan

2 Osnove

U nastavku koristimo standardne pojmove atoma tj. iskaznih slova, literala, klauza, iskaznih formula, veznika $\vee, \wedge, \Rightarrow, \Leftrightarrow, \neg$ kao i logičkih konstanti $\top, \bot$.

2.1 Definicija problema planiranja

Pre nego što formalno definišemo pojam problema planiranja, uvodimo pojam *stanja*, *promenljive stanja* kao i *akcije*.

Definicija 1. Skup atoma (tj. iskaznih slova) A nazivamo skupom *promenljivih stanja*. *Stanje* definišemo kao *valuaciju* nad skupom A, odnosno preslikavanje $s\colon A \to \{0,1\}$. Skup svih stanja obeležavamo sa 2^A. *Akcija (operator)* je uređeni par $o = (p, e)$ gde je:

- p iskazna formula nad A koju nazivamo *preduslov* akcije o
- e skup parova $f \Rightarrow d$ koje nazivamo efektima akcije o, gde je f iskazna formula nad A, a d je skup literala. U slučaju kada $f \neq \top$, tada efekat $f \Rightarrow d$ nazivamo *uslovnim efektom*.

Potrebno je još definisati pojam *izvršavanja* odnosno način kako primenom operatora prelazimo iz jednog stanja u neko drugo stanje.

Definicija 2. *Aktivni efekat* operatora $o = (p, e)$ u stanju s je skup

$$[o]_s = [e]_s = \bigcup\{d | f \Rightarrow d \in e, s \models f\}.$$

Operator je *izvršiv* u stanju s ako $s \models p$ i $[o]_s$ je *konzistentan* skup (tj. ne postoji $a \in A$ tako da $a, \neg a \in [o]_s$). *Izvršavanje* operatora $o = (p, e)$, izvršivog u stanju s, u oznaci $\mathsf{exec}_o(s)$ predstavlja jedinstveno stanje s' takvo da:

- $s'(a) = s(a)$ za $a, \neg a \notin [o]_s$,
- $s'(a) = 1$ za $a \in [o]_s$,
- $s'(a) = 0$ za $\neg a \in [o]_s$.

Za niz operatora $o_1; o_2; \ldots; o_n$ definišemo *niz izvršavanja*

$$\mathsf{exec}_{o_1;o_2;\ldots;o_n}(s) = \mathsf{exec}_{o_n}(\cdots \mathsf{exec}_{o_2}(\mathsf{exec}_{o_1}(s))\cdots).$$

Za operator $o = (p, e)$ i literal l definišemo *preduslov efekta l u odnosu na operator o*

$$\mathsf{EPC}_l(o) = \bigvee\{f | f \Rightarrow d \in e, l \in d\}$$

pri čemu $\bigvee\emptyset = \perp$.

Sada možemo definisati *problem planiranja* kao i *plan izvršavanja* koji predstavlja rešenje tog problema.

Definicija 3. *Problem planiranja* predstavlja uređenu četvorku $\pi = (A, I, \mathcal{O}, \mathcal{G})$ gde je:

- A – promenljive stanja,
- I – početno stanje,
- $\mathcal{O}$ – skup akcija tj. operatora,
- $\mathcal{G}$ – ciljno stanje.

Definicija 4. *Plan izvršavanja* (ili samo *plan*) za problem planiranja π predstavlja niz $\sigma = o_1; o_2; \ldots; o_n$ operatora iz $\mathcal{O}$ takvih da

$$\mathsf{exec}_{o_1;o_2;\ldots;o_n}(I) \models \mathcal{G}$$

Dužina plana $\sigma = o_1; o_2; \ldots; o_n$ u oznaci $\ell(\sigma) = n$ jednaka je dužini niza tj. broju operatora u σ.

Primer. Osoba A se nalazi ispred zaključanih vrata koja se otvaraju ključem. Osoba A ne poseduje ključ, ali ga može nabaviti. Osoba A želi da otključa vrata. Predstavimo ovaj problem kao problem planiranja i definišimo promenljive stanja, stanja, akcije tj. operatore i nađimo jedan plan izvršavanja.

Stanja ovog problema određena su sa dva podatka: da li osoba A poseduje ključ i da li su vrata otključana ili ne. Ova dva podataka modelujemo kao *promenljive stanja* tj. iskazna slova k (od key) i d (od door). Stanje predstavljamo funkcijom koja promenljivim stanja dodeluje istinitosne vrednosti. U ovom primeru imamo četiri stanja:

$$s_1 : \begin{pmatrix} k & d \\ 0 & 0 \end{pmatrix} \quad s_2 : \begin{pmatrix} k & d \\ 0 & 1 \end{pmatrix} \quad s_3 : \begin{pmatrix} k & d \\ 1 & 0 \end{pmatrix} \quad s_4 : \begin{pmatrix} k & d \\ 1 & 1 \end{pmatrix}$$

U stanju s_1 osoba A ne poseduje ključ, a vrata su zaključana. U stanju s_2 osoba A ne poseduje ključ, a vrata su otključana. U stanju s_3 osoba A poseduje ključ, a vrata su zaključana. Konačno, u stanju s_4 osoba A poseduje ključ, a vrata su otključana.

Posmatrajmo akcije odnosno operatore u ovom problemu – ima ih dva. To su operatori $\mathsf{acquire}$ i unlock definisani sa:

$$\mathsf{acquire} = (\neg k, \{\top \Rightarrow k\}) \quad \text{tj.} \quad \mathsf{unlock} = (k, \{\neg d \Rightarrow d\})$$

Operator $\mathsf{acquire}$ govori da osoba A može u bilo kom trenutku da pribavi ključ (ukoliko ga prethodno ne poseduje), dok operacija unlock govori da osoba A ukoliko prethodno kod sebe poseduje ključ može zaključana vrata otključati.

Sada definišemo problem planiranja $\pi = (A, I, \mathcal{O}, \mathcal{G})$ gde je $A = \{k, d\}$, $I = s_1$, $\mathcal{O} = \{\mathsf{acquire}, \mathsf{unlock}\}$, $\mathcal{G} = d$. Plan za ovaj problem je $\sigma = \mathsf{acquire}; \mathsf{unlock}$. Zaista $\mathsf{exec}_{\mathsf{acquire}}(s_1) = s_3$, dok je $\mathsf{exec}_{\mathsf{unlock}}(s_3) = s_4$ i pritom $s_4 \models d = \mathcal{G}$. Na kraju primetimo da (sa ovako definisanim akcijama) nijedno izvršavanje niza neće "proći" kroz stanje s_2.

2.2 Svođenje na SAT

Neka je $\pi = (A, I, \mathcal{O}, \mathcal{G})$ problem planiranja. Definisaćemo kopije skupa promenljivih stanja $A' = \{a' | a \in A\}$ i $A^n = \{a^n | a \in A\}$, $n \geq 0$.

Kopije iz A' koristimo za konstruisanje iskaznih formula τ_o koje odgovaraju operatorima $o \in \mathcal{O}$ – ideja je da promenljive iz A modeluju stanje pre izvršavanja operatora o, dok promenljive iz A' modeluju stanje dobijeno nakon izvršavanja operatora o.

Slično, kopije iz A^n koristimo za konstruisanje iskaznih formula koje odgovaraju izvršavanju niza operatora $o_1; o_2; \ldots; o_n$– ideja je da promenljive iz A^i modeluju stanje nakon izvršavanja i koraka tj. operatora u tom nizu. Za iskaznu formulu $\mathcal{F}$, sa $\mathcal{F}^t$ označavaćemo formulu dobijenu zamenom a sa a^t u $\mathcal{F}$.

Pitanje je kako konstruisati iskaznu formulu τ_o i koja svojstva želimo da ona zadovoljava. Formulu τ_o definišemo nad $A \sqcup A'$. Neka je v proizvoljna valuacija $v \colon A \sqcup A' \to \{0, 1\}$. Posmatrajmo posebno vrednosti promenljivih iz A, a posebno vrednosti promenljivih iz A' tj. neka je $s = v|_A$ i $s' = v|_{A'}$. Valuacije s i s' predstavljaju stanja nad A odnosno A'. Uvedimo oznaku $v = s \sqcup s'$. Dakle, dodela vrednosti promenljivama iz $A \sqcup A'$ generiše dva stanja, jedno definisano nad A (koje predstavlja stanje pre izvršavanja operatora), a drugo definisano nad A' (koje predstavlja stanje dobijeno nakon izvršavanja operatora). Prirodno je očekivati da će vrednost iskazne formule τ_o u valuaciji v biti tačna ako i samo ako su vrednosti promenljivama stanja dodeljene na taj način da valuacije s i s' modeluju "susedna" stanja u odnosu na operator o, a da je pritom operator o izvršiv u stanju s. Iskaznu formulu τ_o možemo posmatrati i kao binarnu relaciju na $2^A \times 2^{A'}$ pri čemu $s \, \tau_0 \, s'$ ako i samo ako s i s' modeluju "susedna" stanja u odnosu na operator o, a da je pritom operator o izvršiv u stanju s.

Imajući sve navedeno u vidu, operator $o = (p, e)$ predstavljamo iskaznom formulom

$$\tau_o = p \wedge \bigwedge_{a \in A} [(\mathsf{EPC}_a(o) \vee (a \wedge \neg \mathsf{EPC}_{\neg a}(o))) \Leftrightarrow a'].$$

Vrednost promenljive a' biće tačna ako i samo ako je ispunjen preduslov efekta za literal a i operator o ili ako je a tačno i nije ispunjen preduslov efekta za literal $\neg a$, a da je pritom ispunjen preduslov operatora. Semantičko opravdanje ovakve konstrukcije iskazne formule τ_o daje sledeća lema.

Lema 1. Neka su s i s^* stanja nad A, a s' stanje nad A' za koja važi $s'(a') = s^*(a)$ za sve $a \in A$. Tada $s \sqcup s' \models \tau_o$ ako i samo ako $s^* = \mathsf{exec}_o(s)$.

Jasno je da u problemu planiranja možemo preći iz jednog stanja s_1 u stanje s_2 ako i samo ako postoji operator $o \in \mathcal{O}$ takav da $s_2 = \mathsf{exec}_o(s_1)$. Ako posmatramo samo konačne skupove operatora $\mathcal{O} = \{o_1, o_2, \ldots, o_n\}$ prethodnu konstataciju možemo raspisati kao disjunkciju:

$$[s_2 = \mathsf{exec}_{o_1}(s_1)] \vee [s_2 = \mathsf{exec}_{o_2}(s_1)] \vee \ldots [s_2 = \mathsf{exec}_{o_n}(s_1)] \equiv \bigvee_{o\in\mathcal{O}} [s_2 = \mathsf{exec}_o(s_1)]$$

S obzirom da smo konstruisali iskaznu formulu kojom modelujemo činjenicu $s_2 = \mathsf{exec}_o(s_1)$, iz gorenavedenog sledi da možemo modelovati činjenicu da iz stanja s_1 možemo doći u stanje s_2 sledećom iskaznom formulom

$$\mathcal{T}(A, A') = \bigvee_{o\in\mathcal{O}} \tau_o.$$

Dakle, vrednost iskazne formule $\mathcal{T}(A, A')$ je tačna u valuaciji v ako i samo ako su vrednosti promenljivama stanja dodeljene na taj način da za valuacije s i s' modeluju "susedna" stanja u odnosu na neki operator o koji je izvršiv u stanju s. Semantičko opravdanje za ovakvu konstrukciju daje nam sledeća lema.

Lema 2. Neka su s i s^* stanja nad A, a s' stanje nad A' za koja važi $s'(a') = s^*(a)$ za svako $a \in A$. Tada $s \sqcup s' \models \mathcal{T}(A, A')$ ako i samo ako postoji operator $o \in \mathcal{O}$ takav da $s' = \mathsf{exec}_o(s)$

Potrebno je da modelujemo i početno stanje problema planiranja I iskaznom formulom ι (primetimo da je završno stanje definisano kao iskazna formula nad skupom promenljivih stanja). S obzirom da je početno stanje I jedna valuacija, konstrukcija iskazne formule ι takve da $v \models \iota$ ako i samo ako $v = I$ je trivijalno:

$$\iota = \bigwedge_{\substack{a\in A\\ I\models a}} a \wedge \bigwedge_{\substack{a\in A\\ I\not\models a}} \neg a$$

Konačno, potrebno je još da modelujemo plan izvršavanja, odnosno rešenje problema planiranja. Iskaznu formulu Σ_t (gde je t dužina plana izvršavanja $\sigma = o_0 o_1 \ldots o_{t-1}$) modelujemo nad skupom promenljivih stanja $\bigsqcup_{i=0}^{t} A^i$ pri čemu važi $s_0 \sqcup s_1 \sqcup \ldots \sqcup s_{t-1} \sqcup s_t \models \Sigma_t$ ako i samo ako je s_0 početno stanje, s_{i-1} i s_i su "susedna" stanja u odnosu na operator o_{i-1} koji je izvršiv u stanju s_{i-1} (za $1 \leq i \leq t$) i $s_t \models \mathcal{G}$. Za problem planiranja $\pi = (A, I, \mathcal{O}, \mathcal{G})$ plan izvršavanja dužine t predstavljamo iskaznom formulom

$$\Sigma_t = \iota^0 \wedge \mathcal{T}(A^0, A^1) \wedge \mathcal{T}(A^1, A^2) \wedge \cdots \wedge \mathcal{T}(A^{t-1}, A^t) \wedge \mathcal{G}^t$$

Ovaj odeljak završavamo teoremom koja daje semantičko opravdanje prethodne konstrukcije.

Teorema 1. Neka je $\pi = (A, I, \mathcal{O}, \mathcal{G})$ problem planiranja. Iskazna formula

$$\iota^0 \wedge \mathcal{T}(A^0, A^1) \wedge \mathcal{T}(A^1, A^2) \wedge \cdots \wedge \mathcal{T}(A^{t-1}, A^t) \wedge \mathcal{G}^t$$

je zadovoljiva ako i samo ako postoji plan izvršavanja σ dužine $\ell(\sigma) = t$ za problem planiranja π.

3 Opis metode

3.1 Opšti metod

Konstrukcijom iskazne formule Σ_t odnosno ispitivanjem njene zadovoljivosti dobijamo odgovor na to da li postoji plan dužine t koji rešava definisani problem planiranja. Naša metoda svodiće se na konstrukciju iskaznih formula $\Sigma_1, \Sigma_2, \ldots, \Sigma_t$ dok ne naiđemo na zadovoljivu formulu Σ_n. U tom trentku se zaustavljamo, prijavljujemo da smo našli plan izvršavanja dužine n i iz dobijene valuacije (koja zadovoljava formulu Σ_n konstruišemo plan izvršavanja $\sigma = o_1; o_2; \ldots; o_n$.

Postavlja se pitanje da li se ovaj postupak završava? S obzirom da postoji 2^n stanja, gde je n broj promenljivih stanja, jasno je da postoji prirodan broj m takav da za svako dostižno stanje s postoji niz izvršavanja dužine najviše m koje počinje u početnom stanju i završava se u s. Ukoliko jedno od dostižnih stanja zadovoljava formulu ciljnog stanja $\mathcal{G}$, tada do tog stanja i odgovarajućeg plana izvršavanja dolazimo u manje od m koraka. U suprotnom, u m koraka posetićemo sva dostižna stanja, od kojih nijedno ne zadovoljava formulu ciljnog stanja $\mathcal{G}$ – u ovom slučaju problem planiranja nema rešenje.

3.2 Opis metode za problem BW

Blocks World (BW) problem. Neka je $\mathsf{B} = \{1, 2, \ldots, n\}$ skup *blokova*. Elemente skupa $\mathsf{T}(\mathsf{B}) = \{(\mathsf{b}_1, \mathsf{b}_2, \ldots, \mathsf{b}_m) | \mathsf{b}_i \neq \mathsf{b}_j \text{ za } i \neq j\}$ nazivamo *kulama*. Skup $S \subset \mathcal{P}(\mathsf{T}(\mathsf{B}))$ nazivamo *stanjem* ako za svako $1 \leq i \leq n$ postoji tačno jedan element $\mathsf{t}_{k_i} \in S$ takav da $(\mathsf{t}_{k_i})_{m_i} = i$ za neko m_i. Stanja S_1 i S_2 su *susedna* ako je ispunjen jedan od sledećih uslova

1. $(\mathsf{b}_{k_1}, \ldots \mathsf{b}_{k_m}) \in S_1$ i $(\mathsf{b}_{k_1}, \ldots \mathsf{b}_{k_{m-1}}), (\mathsf{b}_{k_m}) \in S_2$
2. $(\mathsf{b}_{k_1}, \ldots \mathsf{b}_{k_{m-1}}), (\mathsf{b}_{k_m}) \in S_1$ i $(\mathsf{b}_{k_1}, \ldots \mathsf{b}_{k_m}) \in S_2$
3. $(\mathsf{b}_{k_1}, \ldots \mathsf{b}_{k_m}), (\mathsf{b}_{l_1}, \ldots \mathsf{b}_{l_p}, \mathsf{b}) \in S_1$ i $(\mathsf{b}_{k_1}, \ldots \mathsf{b}_{k_m}, \mathsf{b}), (\mathsf{b}_{l_1}, \ldots \mathsf{b}_{l_p}) \in S_1$

i pritom se stanja S_1 i S_2 poklapaju na ostalim kulama. *BW problem* definisan je kao uređeni par stanja (I, G) gde je I početno, a G ciljno stanje. Rešenje BW problema je *izvršavanje* tj. niz stanja $I = S_0 S_1 \ldots S_k = G$ takav da su stanja S_{i-1} i S_i susedna za svako $1 \leq i \leq n$.

Primer. Za primer sa Slike 1 je:

- $\mathsf{B} = \{1, 2, 3, 4, 5, 6, 7\}$
- $\mathsf{T}(\mathsf{B}) = \{(1), \ldots, (7), (1, 2), \ldots, (7, 6, 5, 4, 3, 2, 1)\}$
- $I = S_0 S_1 \ldots S_n = G$ je jedno izvršavanje za BW problem (I, G) gde je:

$$
\begin{array}{ll}
S_0 = \{(3,2,1),(5,4),(7,6)\}; & S_1 = \{(1),(3,2),(5,4),(7,6)\} \\
S_2 = \{(1),(2),(3),(5,4),(7,6)\}; & S_3 = \{(1),(2),(3,4),(5),(7,6)\} \\
S_4 = \{(1),(2,5),(3,4),(7,6)\}; & S_5 = \{(2,5,1),(3,4),(7,6)\} \\
S_5 = \{(2,5,1),(3,4),(7,6)\}; & S_6 = \{(2,5,1),(3,4,6),(7)\}
\end{array}
$$

Pokazaćemo da se BW problem može definisati i kao problem planiranja. Prethodno opisanom procedurom, rešavanje BW problema svešćemo na ispitivanje zadovoljivosti iskazne formule tj. na SAT problem koji možemo rešiti SAT rešavačem.

3.2.1 Skup promenljivih stanja za problem BW

Jednostavnosti radi, uvedimo još jedan blok $\mathsf{Tbl} = 0$ koji predstavlja sto. Uvedim oznaku $\overline{\mathsf{B}} = \mathsf{B} \cup \{\mathsf{Tbl}\}$. Za svaki blok različit od Tbl uvodimo promenljivu stanja p_i (ukupno n promenljivih) koja označava da se u trenutnom stanju blok i nalazi na stolu Tbl. Nakon toga, za svaki uređeni par različitih blokova $(i, j), i \neq j$ uvodimo promenljivu stanja $\mathsf{q}_{i,j}$ (ukupno $n(n-1)$ promenljivih) koja označava da se u trenutnom stanju blok j nalazi na bloku i u okviru neke od kula. Sada možemo definisati skup promenljivih stanja:

$$A = \{\mathsf{p}_i | i \in \mathsf{B}\} \cup \{\mathsf{q}_{i,j} | i, j \in \mathsf{B}, i \neq j\}$$

Koja ograničenja moraju da važe nad skupom promenljivih stanja u jednom stanju? Jasno je da svaki blok mora biti na stolu ili na nekom drugom bloku:

$$\mathsf{On}(i) = \mathsf{p}_i \vee \bigvee_{\substack{j=1 \\ j \neq i}}^{n} \mathsf{q}_{j,i}$$

Sa druge strane, jedan blok ne može istovremeno biti na stolu i na nekom drugom bloku, niti na dva različita bloka:

$$\mathsf{UniqueOn}(i) = \bigwedge_{\substack{j=1 \\ j \neq i}}^{n} (\neg \mathsf{p}_i \vee \neg \mathsf{q}_{j,i}) \wedge \bigwedge_{\substack{j=1 \\ j \neq i}}^{n} \bigwedge_{\substack{k=1 \\ k \neq i}}^{j-1} (\neg \mathsf{q}_{j,i} \vee \neg \mathsf{q}_{k,i})$$

Potrebno je još obezbediti da se na jednom bloku (koji nije Tbl) ne mogu istovremeno naći dva bloka:

$$\mathsf{UniqueCarries}(i) = \bigwedge_{\substack{j=1 \\ j \neq i}}^{n} \bigwedge_{\substack{k=1 \\ k \neq i}}^{j-1} (\neg \mathsf{q}_{i,k} \vee \neg \mathsf{q}_{i,k})$$

3.2.2 Skup operatora za problem BW

Uvedimo pomoćnu iskaznu formulu za svaki blok i kojom označavamo da je blok i "slobodan" odnosno da se na njemu ne nalazi nijedan drugi blok:

$$\mathsf{Clear}(i) = \bigwedge_{\substack{j=1 \\ j \neq i}}^{n} \neg \mathsf{q}_{i,j}$$

Definišemo operatore $\mathsf{o}_{1,i,j}, \mathsf{o}_{2,i,j}, \mathsf{o}_{3,i,j,k}$ koji odgovaraju susednim stanjima u formulaciji BW problema:

- $\mathsf{o}_{1,i,j} = (\mathsf{q}_{j,i} \wedge \mathsf{Clear}(i), \{\top \Rightarrow \{\neg \mathsf{q}_{j,i}, \mathsf{p}_i\}\})$
- $\mathsf{o}_{2,i,j} = (\mathsf{p}_i \wedge \mathsf{Clear}(i) \wedge \mathsf{Clear}(j), \{\top \Rightarrow \{\mathsf{q}_{j,i}, \neg \mathsf{p}_i\}\})$
- $\mathsf{o}_{3,i,j,k} = (\mathsf{q}_{j,i} \wedge \mathsf{Clear}(i) \wedge \mathsf{Clear}(k), \{\top \Rightarrow \{\neg \mathsf{q}_{j,i}, \mathsf{q}_{k,i}\}\})$

Može se pokazati da navedeni operatori čuvaju korektnost stanja, odnosno u slučaju da iz jednog stanja u kom su za svaki blok ispunjena ograničenja

$\mathsf{On}, \mathsf{UniqueOn}, \mathsf{UniqueCarries}$ izvršavanjem operatora o prelazimo u stanje u kom su za svaki blok ispunjena ograničenja $\mathsf{On}, \mathsf{UniqueOn}, \mathsf{UniqueCarries}$. Ukoliko osiguramo sekvencijalno izvršavanje operatora tj. da u jednom koraku primenjujemo isključivo jedan operator na tačno jedan par odnosno jednu trojku blokova, tada provera pomenuta tri ograničenja nije potrebna.

Međutim, ovo je trivijalno zadovoljeno jer za dato stanje $s \in 2^A$ i dva različita operatora o, o^* važi $\mathsf{exec}_o(s) \neq \mathsf{exec}_{o^*}(s)$ što dalje implicira da ne postoji stanje $s' \in 2^{A'}$ takvo da $s \sqcup s' \models \tau_o, \tau_{o^*}$, odnosno da ukoliko $s \sqcup s' \models \mathcal{T}(A, A')$ tada postoji tačno jedan operator o takav da $s \sqcup s' \models \tau_o$.

3.2.3 Skup stanja za problem BW

U t-tom stanju imamo sledeći skup promenljivih stanja:

$$\mathsf{StateVars}(t) = \bigcup_i \mathsf{p}_i^t \cup \bigcup_{i,j} \mathsf{q}_{i,j}^t$$

Što se tiče početnih i završnih stanja, za primer koji je dat u ovom radu važi $I = \{\mathsf{p}_3, \mathsf{q}_{3,2}, \mathsf{q}_{2,1}, \mathsf{p}_5, \mathsf{q}_{5,4}, \mathsf{p}_7, \mathsf{q}_{7,6}\}$ dok je $\mathcal{G} = \{\mathsf{p}_2, \mathsf{q}_{2,5}, \mathsf{q}_{5,1}, \mathsf{p}_3, \mathsf{q}_{3,4}, \mathsf{q}_{4,6}, \mathsf{p}_7\}$.

4 Implementacija

Implementacija konzolne aplikacije za rešavanje proizvoljnog BW problema oslanja se u potpunosti na Teoremu 1, odnosno konstrukciju iskazne formule Σ_t i njenu transformaciju u konjuktivnu normalnu formu. Ključni korak je transformacija iskazne formule $\mathcal{T}(A, A')$ koja se sastoji od $O(n^3)$ disjunkcija zbog čega, u slučaju ekvivalentne transformacije, dolazimo do kombinatorne eksplozije. Međutim, uvođenjem dodatnih promenljivih stanja, koje nazivamo *selektori operatora*, možemo doći do ekvizadovoljive transformacije na sledeći način

$$\mathcal{T}^*(A, A') = \left(\bigvee_{o \in \mathcal{O}} \mathsf{a}_o \right) \wedge \bigwedge_{o \in \mathcal{O}} (\mathsf{a}_o \Rightarrow \tau_o)$$

uz transformaciju iskazne formle $\mathsf{a}_o \Rightarrow \tau_o$ u konjuktivnu normalnu formu, što će biti prostorne složenosti $O(1)$ za svaki operator, odnosno $O(n^3)$ za celu formulu $\mathcal{T}^*(A, A')$. Algoritam za pronalaženje odgovarajućeg plana je sledeći:

1. Učitaj instancu problema BW tj. početno i ciljno stanje.
2. Transformiši instancu problema BW u instancu problema PP
3. Počevši od $t = 1$ konstruiši formulu Σ_t i ispitaj njenu zadovoljivost transformacijom u KNF i pokretanjem proizvoljnog SAT rešavača.
   (a) Ako je formula nezadovoljiva, uvećaj t i vrati se na korak 3.
   (b) Ako je formula zadovoljiva, idi na korak 4.
4. Transformisati valuaciju koju je vratio SAT rešavač u niz akcija odnosno operatora koji predstavlja plan izvršavanja za instancu PP problema i ispisati odgovarajući plan u terminima instance BW problema.

Napomenimo da svaka instanca BW problema ima rešenje (spustiti sve blokove na sto, pa složiti odgovarajuću kulu), pa se algoritam uvek završava.

Konzolna aplikacija implementirana je u programskom jeziku C++ i sastoji se od tri klase `BlocksWorld`, `PlanningProblem` i `Solver` koje se redom nalaze u prostorima imena `bw`, `pln` i `sat`. Početno, odnosno ciljno stanje učitavamo funckijom `bw::BlocksWorld::load_state` iz datoteka `initial.bw` tj. `goal.bw`. Nakon toga kreiramo objekte klasa `BlocksWorld` i `PlanningProblem`, pri čemu je potrebno izvršiti kodiranje promenljivih stanja:

$$\mathtt{q}_{i,j} \rightarrow (i-1) * |B| + j, \quad \mathtt{p}_i \rightarrow (i-1) * |B| + i$$

U `for` petlji redom kreiramo objekte klase `Solver` koji predstavljaju formule $\Sigma_{\mathtt{t}}$ u KNF, koje transformišemo u DIMACS format i pokrećemo `minisat` rešavač, a kao rezultat dobijamo objekat klase `std::optional<Valuation>`. Ako je taj objekat jednak `std::nullopt`, nismo našli plan dužine `t` i prelazimo u sledeću iteraciju petlje uz inkrementaciju `t++`. U suprotnom, SAT rešavač je prijavio zadovoljivosti i vratio valuaciju koja zadovoljava datu formulu. U tom trenutku prekidamo izvršavanje programa, dok iz odgovarajuće valuacije metodama `pln::PlanninProblem::extract_plan` i `bw::BlocksWorld::plan_to_string` ispisujemo rešenje instance BW problema.

Implementacija se nalazi na repoztorijumu na sledećem linku. Uputstvo za pokretanje aplikacije može se naći u odgovarajućoj `README.md` datoteci.

5 Zaključak

U ovom radu su u odeljcima 1.1, 2.1, 2.2 i 3.1 predstavljeni pojam problema planiranja i jedan od pristupa u rešavanju odnosno pronalaženju problema planiranja primenom SAT rešavača [SAT]. U odeljcima 1.2 i 3.2 prikazan je problem BW [BW] koji se lako može predstaviti kao instanca problema planiranja, dok su u odeljku 4 prikazani detalji programske implementacije algoritma za rešavanje instanci BW problema.

Na ilustrativnom primeru prikazali smo široku primenljivost SAT rešavača na razne probleme u informatici, uz korišćenje raznih tehnika pri samoj implementaciji. Koraci pri daljem radu mogu biti optimizacija algoritma za rešavanje BW problema, ali i proširivanje implementacije za veći broj problema koji se mogu svesti na problem planiranja, a imaju drugačije karakteristike nego BW problem.

Literatura

[SAT] Armin Biere, Marijn Heule, and Hans van Maaren, eds. Handbook of satisfiability. Vol. 185. IOS press, 2009.

[BW] John Slaney and Sylvie Thiébaux. *Blocks World revisited.* Artificial Intelligence (AI). Vol. 125. 2001.