UMJETNA INTELIGENCIJA

Ostali pristupi klasičnom planiranju i analiza različitih pristupa

Domina Hozjan

Siječanj, 2015.

SADRŽAJ:

- ZAKLJUČIVANJE O TRENUTNOM STANJU SVIJETA
- PLANIRANJE POMOĆU PROPOZICIJSKE LOGIKE
- KLASIČNO PLANIRANJE KAO SAT PROBLEM
- KLASIČNO PLANIRANJE KAO CSP PROBLEM
- PLANIRANJE POMOĆU ZAKLJUČAKA U LOGICI PRVOG REDA
- DJELOMIČNO ODREĐENO PLANIRANJE
- ANALIZA PRISTUPA KLASIČNOG PLANIRANJA

ZAKLJUČIVANJE O TRENUTNOM STANJU SVIJETA(1)

- **CILJ U NPR. U WUMPUSOVOM SVIJETU:OMOGUĆITI AGENTU DONOŠENJE ZAKLJUČKA U KOJEM SE POLJU NALAZI, JE LI ODREĐENO POLJE SIGURNO ITD.**
- Prisjetimo se KB-a Wumpusovog svijeta (aksiomi i percepcije). Neki od aksioma su:

$$\neg P_{1,1}$$
$$\neg B_{1,1}$$
$$B_{1,1} \Leftrightarrow P_{1,2} \vee P_{2,1}$$
$$S_{1,1} \Leftrightarrow W_{1,2} \vee W_{2,1}$$
$$W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,3} \vee W_{4,4}$$

- Percepcije: postoje činjenice u KB vrijede samo u određenom vremenskom trenutku – u eksponent pišedmo korak u kojem vrijedi. Npr.:

$$GledaRavno^0$$
$$Smrdi^4$$
$$\neg Smrdi^3$$

takve činjenice koje se mijenjaju u svijetu nazivamo **promjenjive činjenice/okolnosti** (eng. Fluent)

- Povezivanje percepcije o smradu i vjetru u nekom koraku sa pozicijom na kojoj se nalazimo u tom koraku:

$$Pozocija_{x,y}^{t} \Rightarrow (Vjetrovito^t \Leftrightarrow B_{x,y})$$
$$Pozocija_{x,y}^{t} \Rightarrow (Smrdi^t \Leftrightarrow S_{x,y})$$

ZAKLJUČIVANJE O TRENUTNOM STANJU SVIJETA(2)

- Nadalje želimo aksiome koji opisuju kako se okolnosti mijenjaju nakon provođenja neke akcije (**tranzicijski model**), za to su nam potrebni:
  - propozicijski simbol za primjenu akcije (npr. $IdiNaprijed^0$)
  - **Aksiomi efekta:**

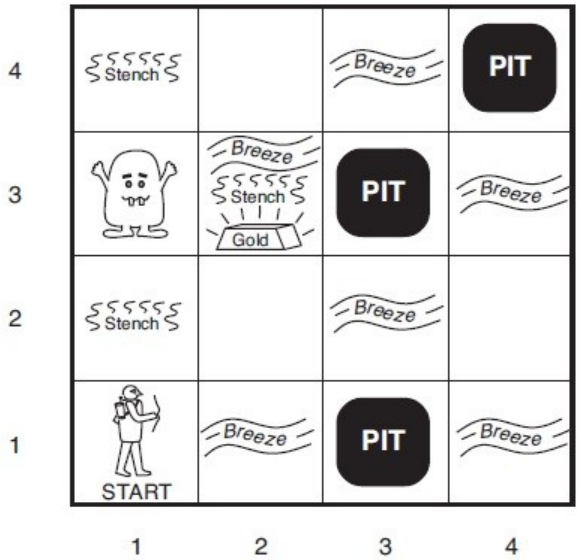
$$Pozicija_{1,1}{}^{0} \wedge GledaPremaIsoku^0 \wedge IdiNaprijed^0 \Rightarrow Pozicija_{2,1}{}^{1} \wedge \neg Pozicija_{1,1}{}^{1}$$

  - **Aksiomi slijedećeg stanja:**

$$\mathbf{F^{t+1} \Leftrightarrow ActionCausesF^t \vee (F^t \wedge \neg ActionCausesNotF^t)}$$

$$Pozicija_{1,1}{}^{t+1} \Leftrightarrow (Pozicija_{1,1} \wedge (\neg IdiNaprije^t \vee ZabioSeUZid^{t+1})) \vee (Pozicija_{1,2}{}^{t} \wedge (Jug^t \wedge IdiNaprijed^t)) \vee (Pozicija_{2,1}{}^{t} \wedge (Zapad^t \wedge IdiNaprijed^t))$$

ZAKLJUČIVANJE O TRENUTNOM STANJU SVIJETA(3)



$$\neg Stench^0 \wedge \neg Breeze^0 \wedge \neg Glitter^0 \wedge \neg Bump^0 \wedge \neg Scream^0 \; ; \; Forward^0$$
$$\neg Stench^1 \wedge Breeze^1 \wedge \neg Glitter^1 \wedge \neg Bump^1 \wedge \neg Scream^1 \; ; \; TurnRight^1$$
$$\neg Stench^2 \wedge Breeze^2 \wedge \neg Glitter^2 \wedge \neg Bump^2 \wedge \neg Scream^2 \; ; \; TurnRight^2$$
$$\neg Stench^3 \wedge Breeze^3 \wedge \neg Glitter^3 \wedge \neg Bump^3 \wedge \neg Scream^3 \; ; \; Forward^3$$
$$\neg Stench^4 \wedge \neg Breeze^4 \wedge \neg Glitter^4 \wedge \neg Bump^4 \wedge \neg Scream^4 \; ; \; TurnRight^4$$
$$\neg Stench^5 \wedge \neg Breeze^5 \wedge \neg Glitter^5 \wedge \neg Bump^5 \wedge \neg Scream^5 \; ; \; Forward^5$$
$$Stench^6 \wedge \neg Breeze^6 \wedge \neg Glitter^6 \wedge \neg Bump^6 \wedge \neg Scream^6$$

$ASK(KB, Pozicija_{1,2}^{6}) = true$

$ASK(KB, Wumpus_{1,3}) = true$

$ASK(KB, Ponor_{3,1}) = true$

$OK_{x,y}^{t} \Leftrightarrow \neg Ponor_{x,y} \wedge \neg(Wumpus_{x,y} \wedge WumpusŽiv^{t})$

$ASK(KB, OK_{2,2}^{6}) = true$

PLANIRANJE POMOĆU PROPOZICIJSKE LOGIKE (1)

```
function SATPLAN(init, transition, goal, T_max) returns solution or failure
  inputs: init, transition, goal, constitute a description of the problem
          T_max, an upper limit for plan length

  for t = 0 to T_max do
    cnf ← TRANSLATE-TO-SAT(init, transition, goal, t)
    model ← SAT-SOLVER(cnf)
    if model is not null then
      return EXTRACT-SOLUTION(model)
  return failure
```

KONSTRUKCIJA REČENICE KOJA SADŽI:

- PočetnoStanje^0
- Aksiom slijedećeg stanja za svaku moguću akciju u svakom vremenskom koraku
- Činjenicu da je ciljno stanje

  ImamZlato^t ∧ IzašaoSamVan^t

IZVLAČENJE IZ MODELA SVIH VARIJABLI KOJE PREDSTAVLJAJU NEKU AKCIJU I OZNAČENE SU KAO *true*

PLANIRANJE POMOĆU PROPOZICIJSKE LOGIKE (2)

PROBLEMI:	RJEŠENJE:
KB: $Pozicija_{1,1}^{0}$, CILJ: $Pozicija_{2,1}^{1}$ SatPlan rješenja: $[IdiNaprijen^{0}]$ $[Pucaj^{0}]$	UMETANJE U KB ČINJENICE DA AGENT U POČETNOM STANJU MOŽE BITI SAMO NA JEDNOJ POZICIJI
SatPlan PRONALAZI I RJEŠENJA S NEMOGUĆIM AKCIJAMA (npr.: PUCANJE BEZ POSJEDOVANJA STRELICE)	AKSIOM SLIJEDEĆEG STANJA NIJE DOVOLJAN. UVODIMO PREDUVJETNE AKSIOME (npr.: $Pucaj^{t} \Rightarrow PosjedujemStrelicu^{t}$)
KREIRANJE VIŠESTRUKIH AKCIJA U ISTOM VREMENSKOM KORAKU (npr.:$IdiNaprijen^{0}$ I $Pucaj^{0}$)	UVODIMO AKSIOME ISKLJUČENJA AKCIJA, ZA SVAKI PAR AKCIJA: $\neg Akcija_{i}^{t} \vee \neg Akcija_{j}^{t}$

KLASIČNO PLANIRANJE KAO SAT PROBLEM

- CILJ: PREVESTI PDDL OPIS U FORMU KOJA MOŽE BITI OBRAĐENA POMOĆU SatPlan-a
- Koraci:

1. Supstitucija konstanti za svaku varijablu u **akcijskoj shemi**

2. **Defininicija početnog stanja** (za svaku promjenjivu činjenicu iz početnog stanja umetnuti Okolnost^0 i ¬Okolnost za one koji se se spominju)

3. **Propozicioniranje ciljnog stanja** (za svaku varijablu u ciljnom stanju zamijeniti svaki literal koji sadrži tu varijavlu sa disjunkcijom literala koji sadrže sve moguće konstante)

4. Dodavanje **aksioma slijedećeg stanja** za svaku okolnost

5. Dodavanje **preduvjetnih aksioma** za svaku akciju

6. Dodavanje **aksioma isključenja akcije**

KLASIČNO PLANIRANJE KAO CSP PROBLEM

- **Definicija CSP problema:**

  -Skup varijabli $\mathcal{X}$ $\{X_1,...,X_n\}$

  -Skup domena $\mathcal{D}$ $\{D_1,...,D_n\}$- D_i sadrži skup dozvojenih vrijednost $\{v_1,...,v_k\}$ za varijablu X_i

  -Skup ograničenja $\mathcal{C}$ - specificira dozvoljenu kombinaciju varijabli, svako ograničenje sadrži par <područje,relacije> (relacije određuju koje vrijednosti mogu poprimiti varijable iz područja)

- Rješenje CSP problema temelji se na dodjeljivanju vrijednosti svim varijablama

- RAZLIKE U ODNOSU NA PREVOĐENJE U SAT PROBLEM:

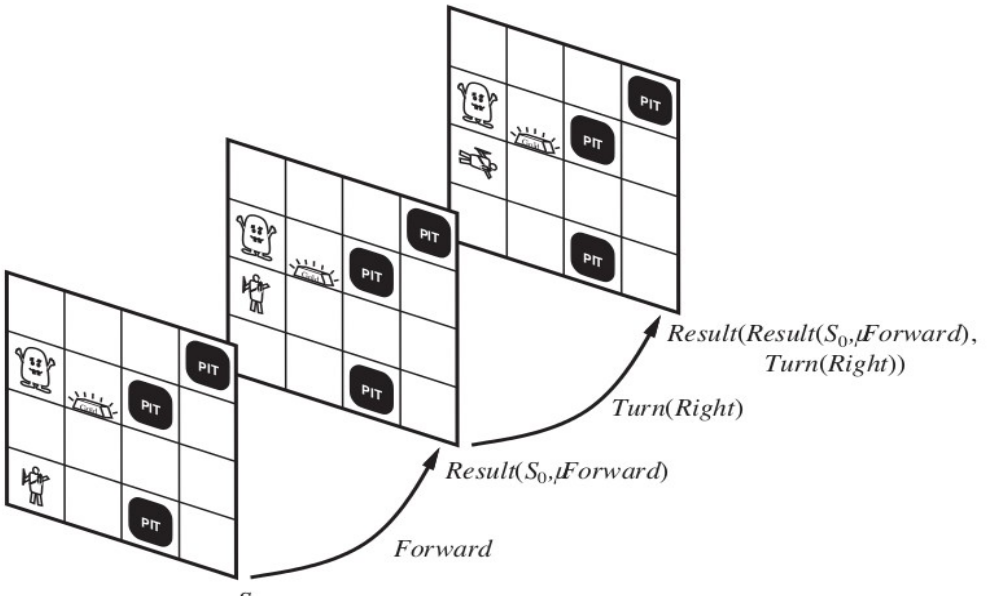
  -U svakom vremenskom koraku trebamo samo jednu varijablu $Akcija^t$ (zbog def. Domene)

  -Ne trebamo više aksiom isključenja akcija

PLANIRANJE POMOĆU ZAKLJUČAKA U LOGICI PRVOG REDA(1)

- RJEŠENJE ZA NEDOSTKE U PROPOZICIJSKOJ LOGICI I PDDL JEZIKU – UVOĐENJE **SITUACIJA:**

  -POČETNO STANJE JE SITUACIJA, AKO JE *s* SITUACIJA, A *a* AKCIJA Rezultat(*s*,*a*) JE SITUACIJA I NEMA DRUGIH SITUACIJA



PLANIRANJE POMOĆU ZAKLJUČAKA U LOGICI PRVOG REDA(2)

- Funkcijske (p=Pozicija(x,s)) i relacijske (NalazimSeU(x,p,s)) okolnosti

- Preduvjete akcija izražavamo pomoću **aksioma mogućnosti** oblika
  R(s)⇒Mogućnost(a,s)

- Svaku okolnost opisujemo **aksimom slijedećeg stanja** oblika **Mogućnost(a,s)⇒**
  (Okolnost_je_istinita_nakon_akcije ⇔ Okolnost_je_rezultat_akcije ∨
  (Okolnost_je_postojala_prije_akcije ∧
  nije_se_dogodija_akcija_koja_bi_uzrokovala_NeOkolnost))

[npr.: Mogućnost(a,s) ⇒ (Posjeduje(Agent,zlato,Rezultat(a,s)) ⇔ a=UzmiZlato ∨ (Posjeduje(Agent,zlato,s) ∧ a≠BaciZlato)]

- Da bi agent mogao zaključiti a≠BaciZlato potrebni su nam aksiomi jedinstvene akcije

$A_i(x,...) \neq A_j(y,...)$ i $A_i(x_1, ..., x_n) = A_i(y_1, ..., y_n) \Leftrightarrow x_1 = y_1 \wedge ... \wedge x_n = y_n$

- Rješenje je situacija koja zadovoljava cilj

DIJELOMIČNO ODREĐENO PLANIRANJE (1)

- Plan sadrži akcije i skup ograničenja *Prije*(a_i,a_j)
- Krećemo sa "praznim planom", tražimo mu mane te potom radimo korekcije tih mana
- Velika popularnost ovog pristupa u 80.-im i 90.-im godinama, a 2000.-ih ga zamjenjuje forward search planiranje s heuristkom
- Danas se koristi u problemima raspoređivanja i u osmišljavanju plana za koji je važno da je razumljiv ljudima

DIJELOMIČNO ODREĐENO PLANIRANJE (2) - PRIMJER

START

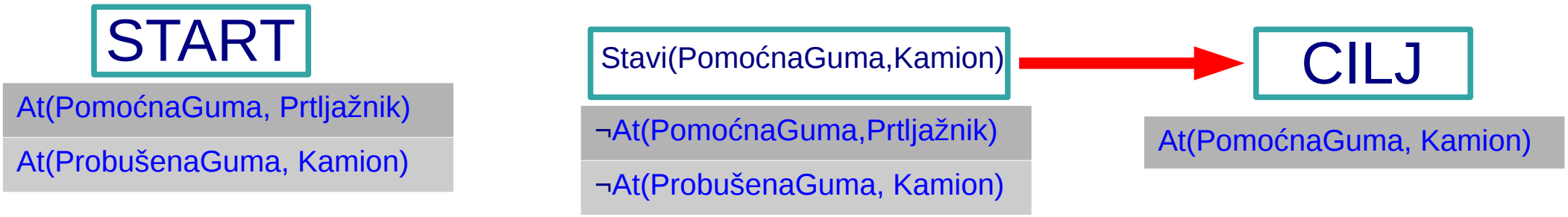
At(PomoćnaGuma, Prtljažnik)

At(ProbušenaGuma, Kamion)

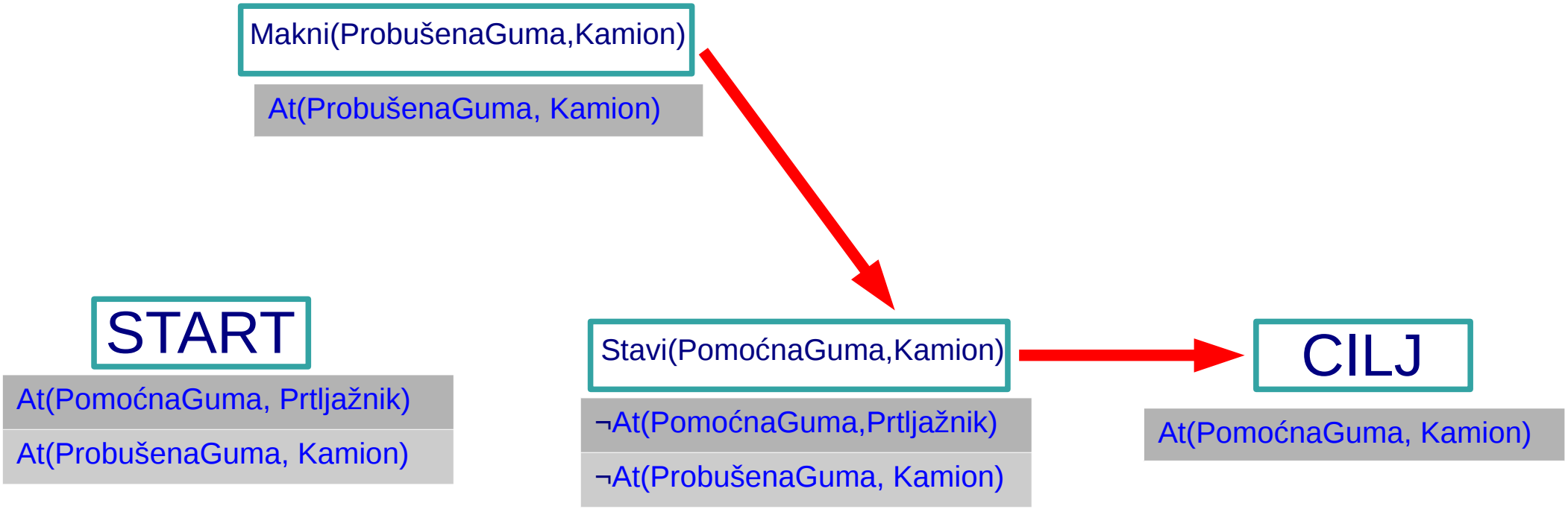
CILJ

At(PomoćnaGuma, Kamion)

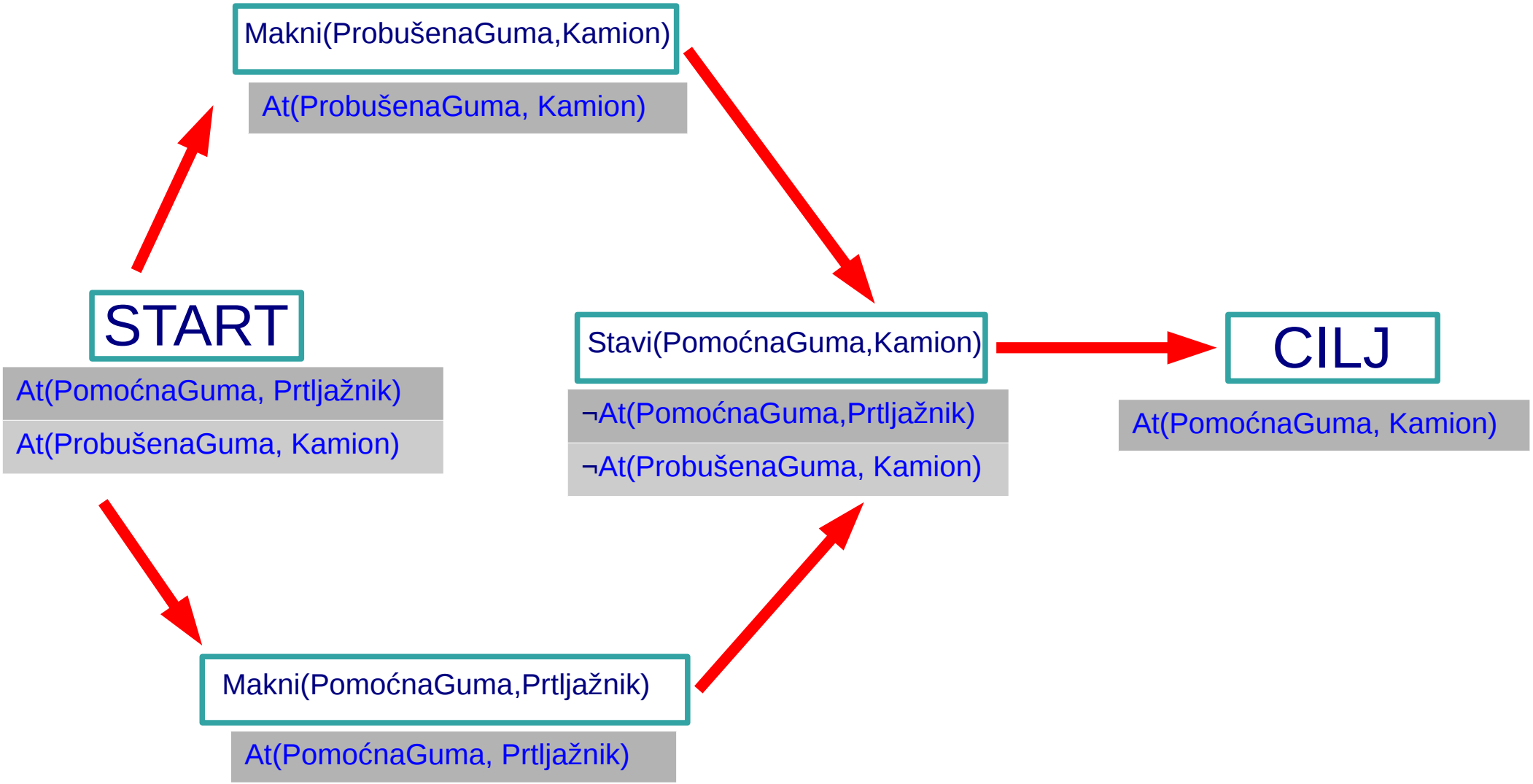
DIJELOMIČNO ODREĐENO PLANIRANJE (2) - PRIMJER



DIJELOMIČNO ODREĐENO PLANIRANJE (2) - PRIMJER

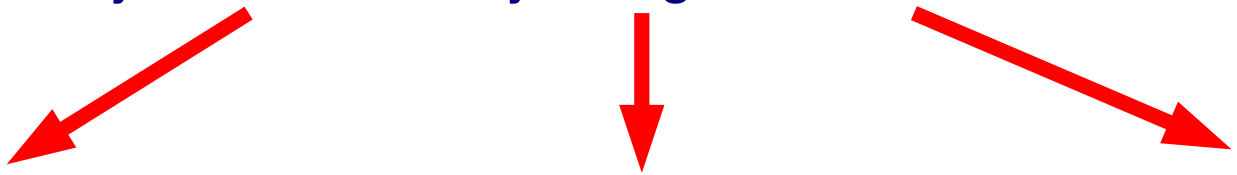


DIJELOMIČNO ODREĐENO PLANIRANJE (2) - PRIMJER



ANALIZA PRISTUPA KLASIČNOG PLANIRANJA

- Planiranje je važan dio UI – kombinira pretraživanje i logiku
- Nažalost neznamo još koje je pristupe najbolje koristiti u kojim vrstama problema
- Najbrže se rješavaju potpuno razgranati problemi, ali zavisnosti među akcijama narušavaju razgranatost



GraphPlan koristi mutex kako bi otkrio zavisnosti

SatPlan koristi CNF

Forward search traži zavisnosti pomoću heuristike

- Slijed podciljeva – planiranje može u određenom rasporedu rješavati podciljeve bez vraćanja na prethodne (rješavanje problema dolje prema gore)

HVALA NA PAŽNJI :)