Wrocław University of Technology

Planowanie i reprezentacja STRIPS

Halina Kwaśnicka
Politechnika Wrocławska
Halina.kwasnicka@pwr.edu.pl

Planowanie

- **Planowanie** - technika rozwiązywania problemów, obejmuje określenie sekwencji akcji, które spowodują przejście systemu ze stanu początkowego do celowego
- Podobieństwa planowania i wnioskowania na bazie reguł:
  przy odpowiednich reprezentacjach oba problemy obejmują dopasowanie w celu określenia stosowalnych reguł lub operacji
- Różnice:
  - reprezentacja dla planowania jest zwykle inna i bardziej złożona
  - planowanie prawie zawsze wymaga r*evocable* strategii sterowania (**noncommutative system**)
  - prawdopodobna produkcja konfliktowych subcelów z powodu istnienia interakcji między operatorami

Planowanie – definicja problemu

- Dane: ???
- Szukamy: ???

- „Planning agent” różni się od „problem-solving agent”:
  - reprezentacją celów, stanów, akcji
  - stosowaniem bezpośredniej, logicznej reprezentacji
  - sposobem poszukiwania rozwiązań

Planning-Based podejście do sterowania robotem

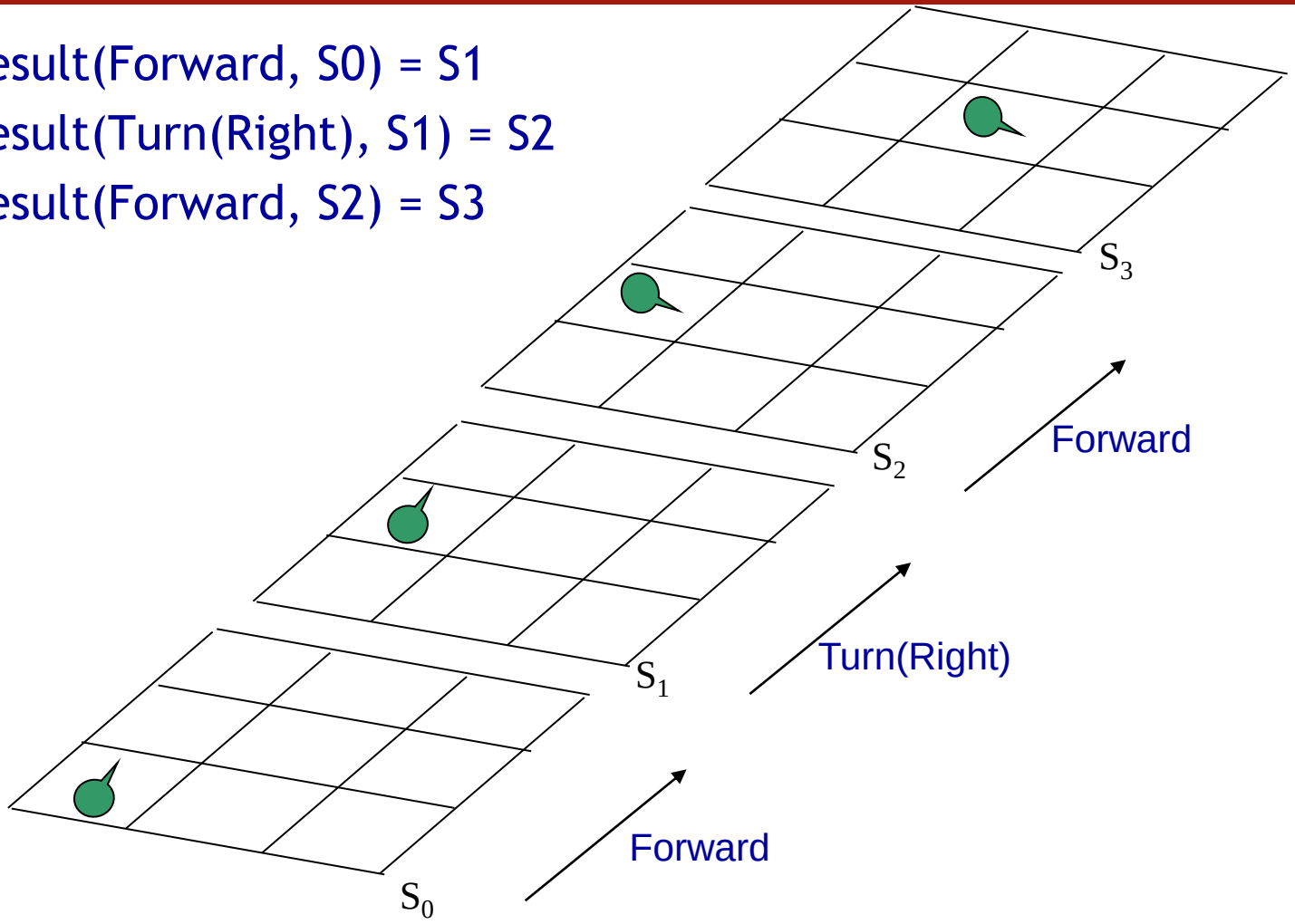
- Zadanie planera: wygenerowanie celu do osiągnięcia, konstrukcja planu – osiągnięcie celu z aktualnego stanu
- Potrzebne reprezentacje
  - reprezentacja akcji: programy generujące opisy kolejnych stanów, definicje warunków wstępnych i efektów
  - reprezentacja stanów: struktury danych opisujące bieżące sytuacje
  - reprezentacja celów: co ma być osiągnięte
  - reprezentacja planów: rozwiązanie jest sekwencją akcji

Przetwarzanie pozycji/stanów/sytuacji

- Obliczenia pozycji/sytuacji (stanu)
  - odnosi się do konkretnego sposobu opisu zmian w logice pierwszego rzędu
  - rozpatruje świat jako składający się z sekwencji sytuacji (stanów), każda jest obserwacją stanu świata (wyizolowaną)
- Wszystkie relacje/własności mogą się zmieniać w czasie
  - Specyfikuje dodatkowo sytuacje jako argumenty predykatów
  - zamiast At(agent,location) wskazuje At(agent, location, S_i) – S_i jest specyficzną sytuacją lub punktem w czasie
- Reprezentuje, jak świat się zmienia z jednej sytuacji (stanu) do drugiej:
  - Result(Forward, S_0) = S_1
  - Result(Turn(Right), S_1) = S_2
  - Results(Forward, S_2) = S_3

Przykład świata reprezentowanego przez Situation Calculus

- Result(Forward, S0) = S1
- Result(Turn(Right), S1) = S2
- Result(Forward, S2) = S3



STRIPS -Stanford Research Institute Problem Solver

- STRIPS - język:
  - klasyczne podejście, najczęściej używane
  - umożliwia efektywne algorytmy planowania
  - ekspresywny w obliczeniach stanów
- STRIPS – stany i cele
  - **stan** – koniunkcja podstawowych literałów, **bez zmiennych** (nie ma funkcji, predykaty ze stałymi symbolami, możliwe negacje)
    - przykład: At(Home) Have(Milk)
    - powszechne założenie: jeśli w opisie stanu nie jest wymieniony literał jako pozytywny (*True*), to przyjmuje się jako *False* (domkniętość świata)
  - Cele: koniunkcja literałów, które **mogą zawierać zmienne**
    - przykład: At(x) Sels(x,Milk)
    - powszechne założenie: zmienne mają kwantyfikatory egzystencjalne

STRIPS reprezentacja akcji (operatorów)

- Operator (akcja) w STRIPS składa się z:
  - Własność (opis) akcji (1): nazwa i opis znaczenia
  - warunki wstępne (precondition) (2): koniunkcja prawdziwych literałów (atomów)
  - efekt: koniunkcja literałów (prawdziwych lub negatywnych) opisujących jak zmieni się sytuacja po wykonaniu akcji (operatora)
  - Reprezentacja efektów:
    - ADD lista (3)
    - DELETE lista (4)

STRIPS - reprezentacja planu

Plan – struktura danych składająca się z:

- zbioru kroków planu – każdy krok jest jednym z dostępnych operatorów
- zbioru ograniczeń na porządek operatorów: ograniczenia porządku to specyfikacja, że jedna z akcji musi wystąpić przed inną
- zbioru ograniczeń na wiązanie zmiennych – są one postaci $v=x$, gdzie v to zmienna w pewnym kroku a x jest zmienną lub stałą
- zbioru związków (połączeń) przyczynowych: opisują cele poszczególnych kroków, np. celem S_i jest osiągnięcie warunku c stanu S_j

Zasady generowania planów w STRIPS

- Stosuj zasadę najmniejszego ‘kosztu’
  - wybieraj tylko to, co jest teraz ważne – pozostaw resztę na później
  - pozostaw porządek kroków (sekwencję) nieokreślony jeśli jest to teraz nieistotne
  - plan, w którym część kroków jest uporządkowanych a pozostałe nie, to *częściowo uporządkowany* plan (partial order plan) w odróżnieniu od planu całkowicie uporządkowanego (total order plan)
- Ukonkretniaj: przypisz zmiennym stałe wartości

  przykład: At(Robot,x) At(Robot, Home)
- W pełni ukonkretnione plany: każda zmienna ma przypisaną wartość

Rozwiązanie w STRIPS

Rozwiązanie: plan realizowalny, gwarantujący osiągnięcie celu

- Łatwo zagwarantować: całkowicie uporządkowany plan, bez zmiennych
- Sposób nie jest satysfakcjonujący, bo:
  - łatwiej ‘planerowi’ zwrócić częściowo uporządkowany plan niż arbitralnie wybierać spośród alternatyw całkowitego uporządkowania
  - część agentów może wykonywać akcje równolegle
  - chcielibyśmy móc łatwiej zarządzać wykonaniem planu w przypadku powstałych później innych ograniczeń
- Dlatego rozwiązanie jest kompletnym i spójnym planem
  - **Kompletny plan** (Complete plan):
    - każdy warunek wstępny na każdym kroku jest osiągalny przez inne kroki
    - warunki wstępne są produkowane przez inne kroki dopiero wtedy, gdy są potrzebne
  - **Spójny plan** (Consistent):
    - nie zawierający sprzeczności w porządku akcji i wartościach zmiennych

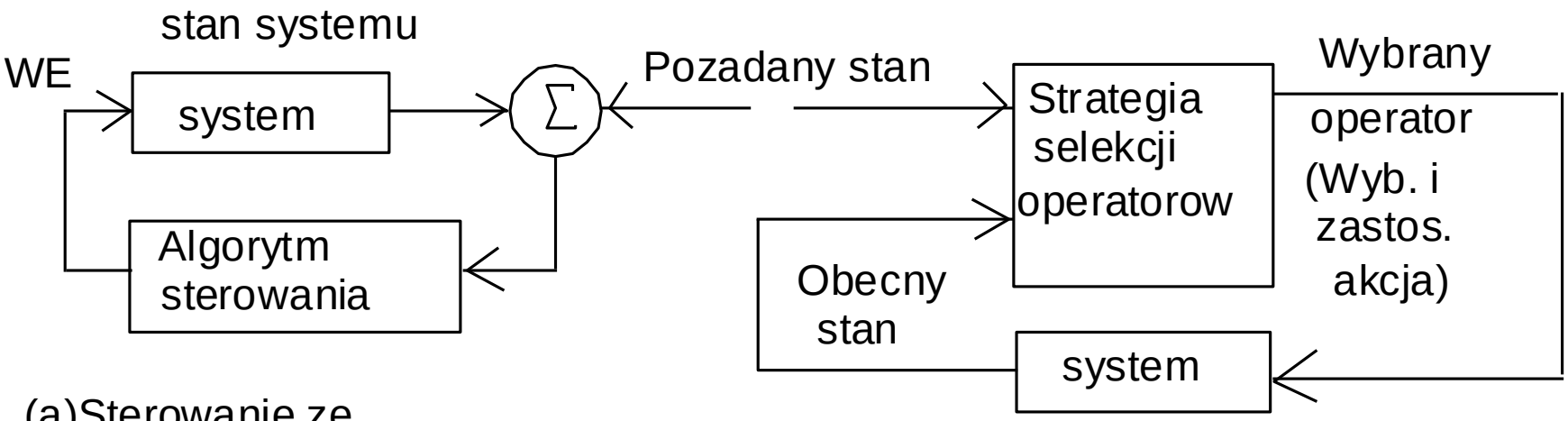
Kilka uwag o planowaniu

- w planowaniu występuje wszechobecny w AI ‘search problem’ (selekcja operatorów)
  - **problem konfliktowych subcelów**
  - do osiągnięcia celu potrzeba koniunkcji subcelów, osiąganych przez subplany
  - te subplany często prowadzą do konfliktów w pożądanym stanie, np. (on A B) i (on B A)
  - takie efekty pojawiają się często przy planowaniu równoległym
- Dla *n* operatorów istnieje *n*! możliwych "uporządkowań" (potencjalna złożoność obliczeniowa)

Przypomnijmy: w STRIPS każdy operator opisuje się w 4 punktach:

1. Własność akcji, jest zdaniem mówiącym, co operator realizuje (ważne dla rozumienia i dokumentacji)
2. Zbiór warunków wstępnych, w formie klauzul, który musi być TRUE zanim operator będzie zastosowany
3. Zbiór 'clause-based' starych faktów, które będą usunięte z opisu aktualnego systemu po zastosowaniu operatora (nie będą już prawdziwe)
4. Zbiór 'clause-based' nowych faktów, które będą dodane do opisu stanu systemu

Sterowanie ze sprzężeniem zwrotnym a system produkcji

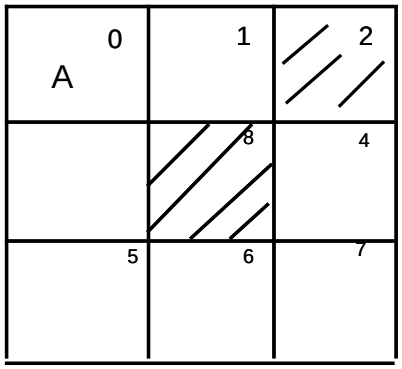


(a)Sterowanie ze sprzezeniem zwrotnym.

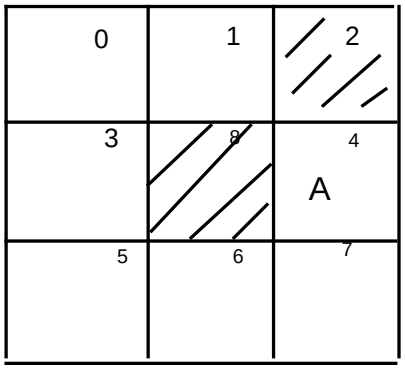
(b)Struktura Systemu Produkcji.

Rys. 14.6.

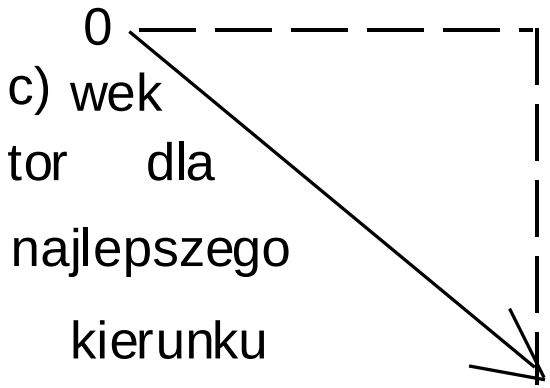
Trudności w selekcji operatorów



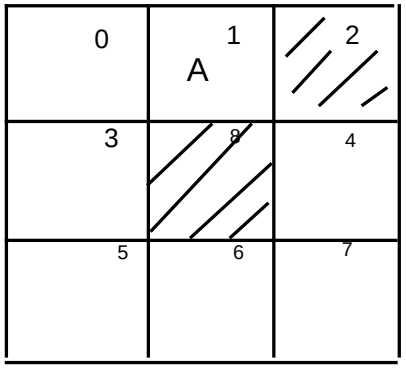
a)stan pocz.



b) stan celowy



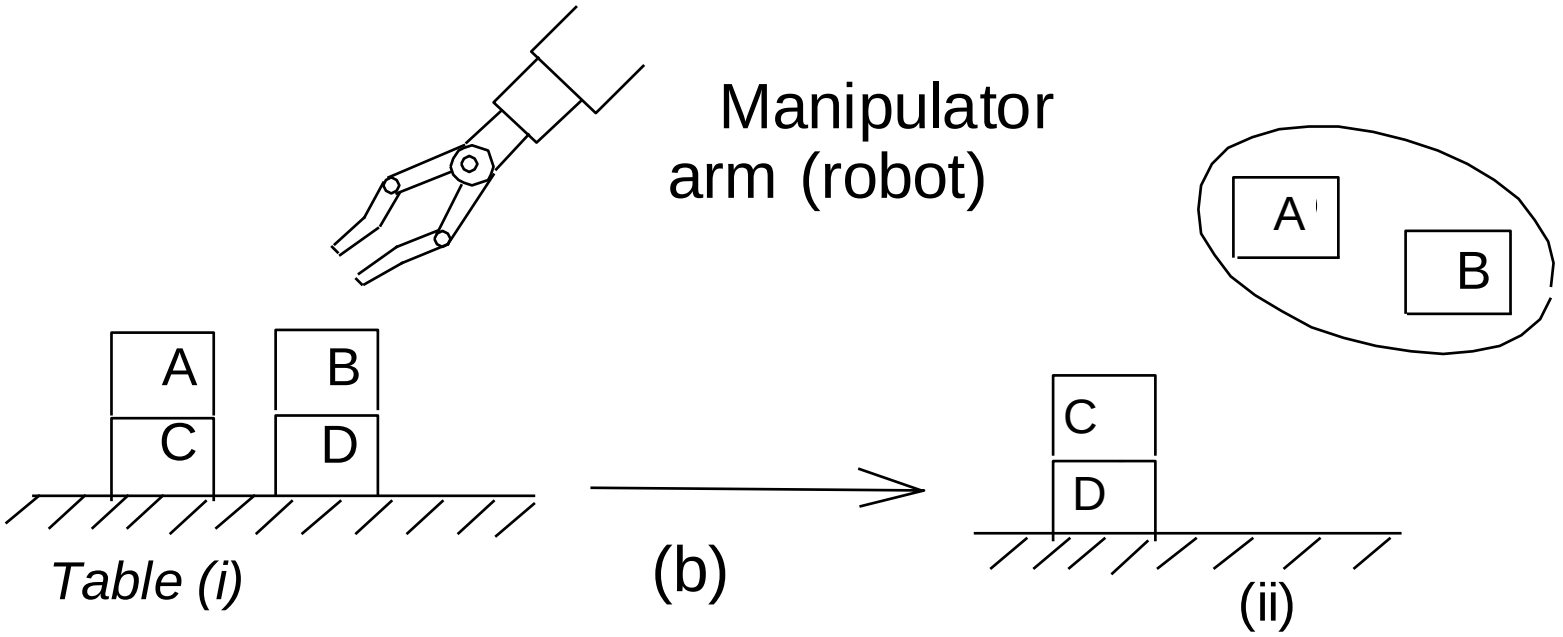
c) wektor dla najlepszego kierunku



d)wynikowy stan -backtracking bedzie konieczny

- Trudno ‘widzieć’ cel lub estymować odległość celu od aktualnego (pożądane w selekcji operatorów)
- Plany mogą się wydawać sprzeczne z intuicją (cofanie się piłkarza spod bramki by uniknąć odparcia obrony)
- Lokalne planowanie (cel na teraz) może być nieefektywne (rys.)

Przykład – opis świata



Stan początkowy i końcowy

Opis stanów – początkowy i końcowy

Stan początkowy S_i :

- (on A C) Klauzula (on x y) - blok x jest na bloku y
- (on B D) Predykat (clear x) - na bloku x nic nie ma
- (ontable C) Predykat (ontable x) – blok x jest na stole
- (ontable D)
- (clear A) predykat (clear x) na bloku x nic nie ma
- (clear B)
- (empty) ramię manipulatora jest aktualnie puste

Stan celowy S_g :

- (on C D)
- (clear C)
- ontable D)

brak informacji o blokach A i B, nie jest wymagane nic odnośnie ramienia

Block - world - specyfikacja operatorów

A: Table-based block manipulation operators

1. **(pickup x)**

   action: pickup block x (from the table)
   preconditions: (ontable x), (clear x), (empty)
   delete facts: (ontable x), (clear x), (empty)
   add facts: (hold x)

2. **(putdown x)**

   action: puts block x down (on the table).
   preconditions: (hold x).
   delete facts: (hold x).
   add facts: (ontable x), (clear x), (empty).

C.d. - operatory

B: Block-based block manipulation operators

3. **(takeoff x y)**

action: takes block x off block y.
precondiotion: (on x y), (clear x), (empty).
delete fakts: (on x y), (clear x), (empty).
add fakts: (clear y), (hold x).

4. **(puton x y)**

action: puts block x on top of block y.
preconditions: (hold x), (clear y).
delete facts: (hold x), (clear y).
add facts: (on x y), (clear x), (empty)

GAT (generate-and-test) - ślepa siła, choć systematycznie

- generowanie wszystkich możliwych rozwiązań i sprawdzanie ich wykonalności
- użyteczny, gdy liczba możliwych rozwiązań jest mała (eksplozja kombinatoryczna)
- możliwe jest stosowanie 'forward state propagation ' FSP i 'backward state propagation' BST
- łatwo jest wygenerować zamknięte cykle
- dla identyfikacji cykli wymagane jest trzymanie śladu wcześniej osiągniętych stanów
- jak i w którym miejscu przerwać cykl?
- niestety, poza prostymi przypadkami, nie jest możliwe ogólne wyznaczenie rozwiązania tego problemu

Forward State Propagation (FSP) bez heurystyki (tzn. GAT)

0(a) Reprezentuj stan początkowy S_i i stan docelowy S_g

0(b) Reprezentuj operatory (O_i)

0(c) Ustal aktualny stan jako stan początkowy S_i

0(d) Zatrzymaj jeśli $S_g \subseteq S_i$

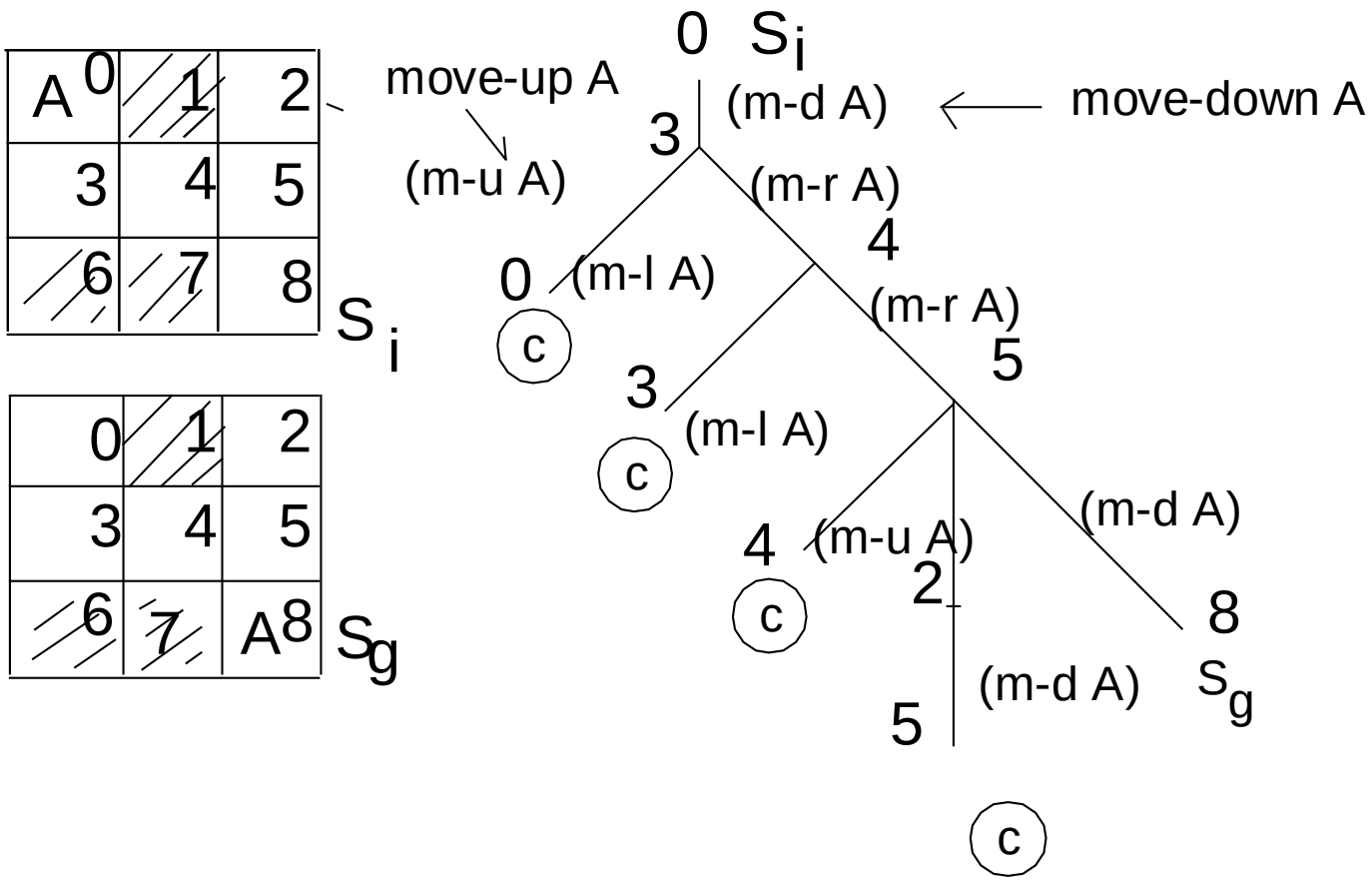
1. Sformułuj L_o - O_i których warunki wstępne są spełnione w S_i
   - jeśli L jest puste trzeba zawrócić
   - jeśli nie ma alternatyw - planu nie można znaleźć
2. Wybierz jeden operator $O_i \in L_o$ i (tymczasowo) sformułuj S_{i+1} jako wynik zastosowania O_i do S_i

3a). Sprawdź czy są konfliktowe subcele w S_{i+1}
   - jeśli istnieją - zrewiduj selekcję operatora

3(b). Zatrzymaj dla sprawdzenia cyklu
   - a) jeśli istnieje cykl, zawróć i wybierz inny operator
   - b) jeśli nie ma cyklu, przejdź do kroku 4

4. Sprawdź, czy S_g jest zawarty w S_{i+1} - jeśli tak - SUKCES, jeśli nie - podstaw $S_i \leftarrow S_{i+1}$ i przejdź do kroku 1.

Przykład FSP - GAT



Backward State Propagation (BSP) (1)

L_C - lista klauzul w S_g, których nie ma w S_i

jeśli S_g zawarty jest w S_i to wtedy L_c pusta

1. Sformułuj L_c dla danego S_i, jeśli L_c pusta to STOP
2. Wybierz jedną klauzulą C_i z L_c
3. Określ listę operatorów, L_o takich, które mają klauzulę C_i w ich części Add

- Ograniczenia elementów L_o:
  - a) Zastosowanie operatorów nie powinno dać konfliktów (sprzeczności w stanie systemu)
  - b) Operatory nie powinny wymagać konfliktu w stanie systemu faktów wymaganych przed zastosowaniem operatora (ukonkretnianie zmiennych może dać (on A A))
- 4. Wybierz jeden operator, O_i z L_o. Kroki 1-3 obejmują selekcję (search)

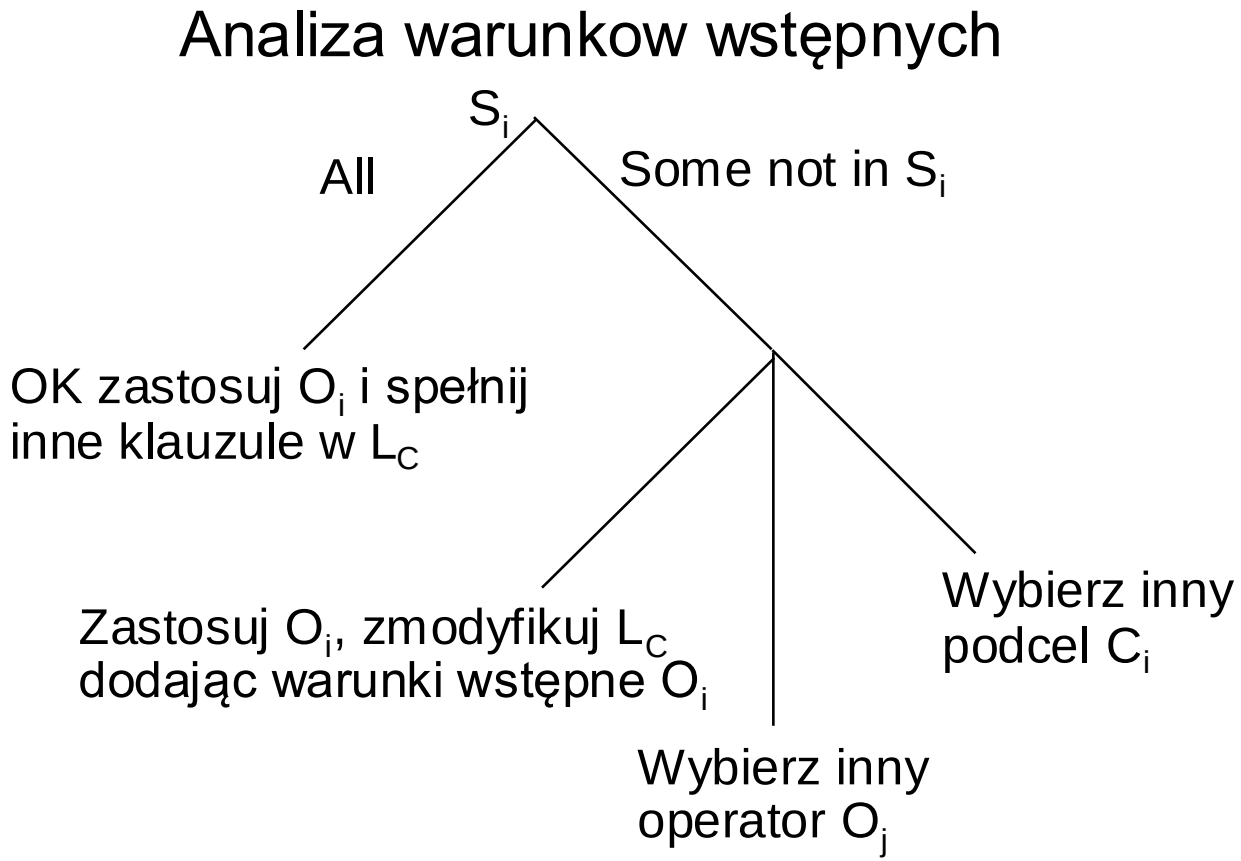
Backward State Propagation (BSP) (2)

5. Określ, które - jeśli są - warunki wstępne operatora O_i wybranego w kroku 4 są spełnione w S_i
   - IF wszystkie warunki są spełnione
   - THEN zastosuj O_i, zmodyfikuj L_c i przejdź do kroku 2 (by spełnić pozostałe podcele)
   - ELSE {inna decyzja jest wymagana}
   - IF inny operator w L_o jest możliwy THEN

     a)wybierz inny operator O_j i powtórz powyższą procedurę
   - OR {inny punkt decyzyjny algorytmu}

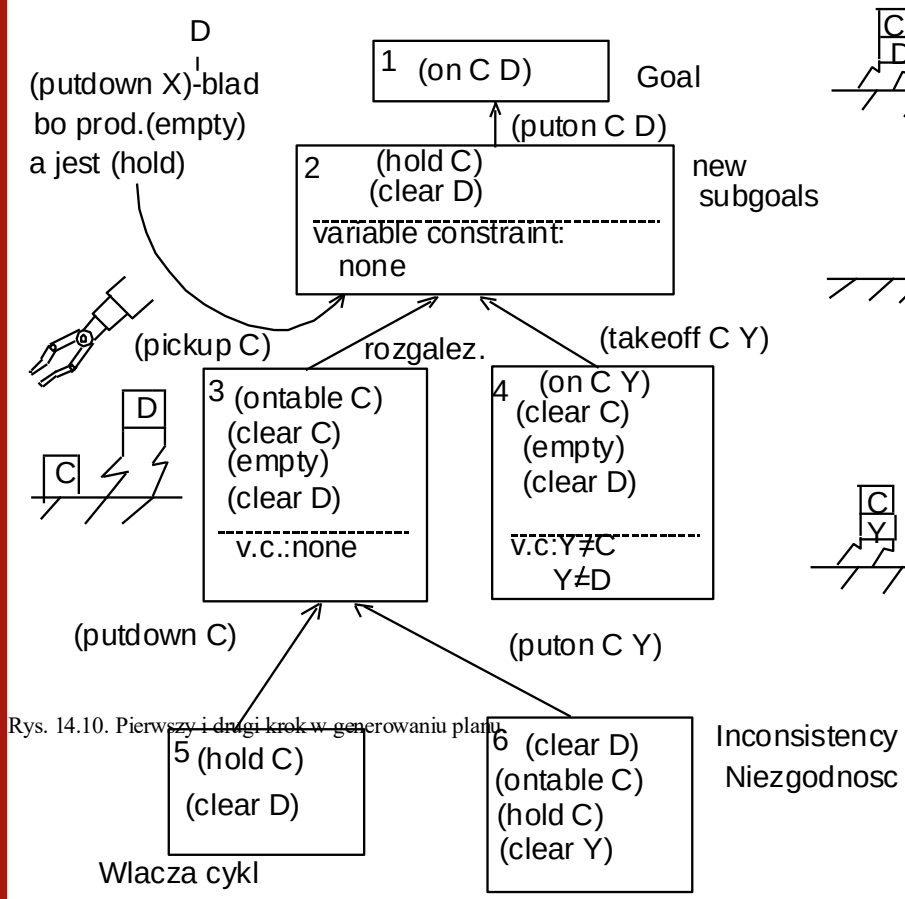
     b)zdecyduj o użyciu wybranego operatora O_i i dodaj niespełnione warunki wstępne O_i jako inne, niespełnione podcele do L_c
   - znowu konfliktowe podcele i cykle wymagają zatrzymania (Rysunek ilustruje możliwe alternatywy w kroku 5)
- 6. Przejdź do kroku 2.

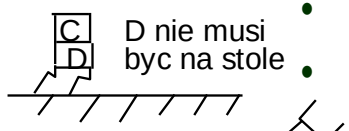
Analiza warunków wstępnych - punkt 5.

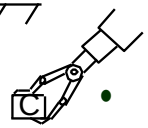


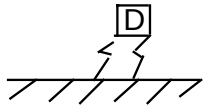
Rys. 14.9. Check for cykles and conflicting subgoals

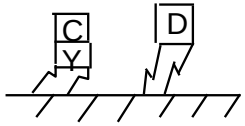
Pierwsze kroki BSP







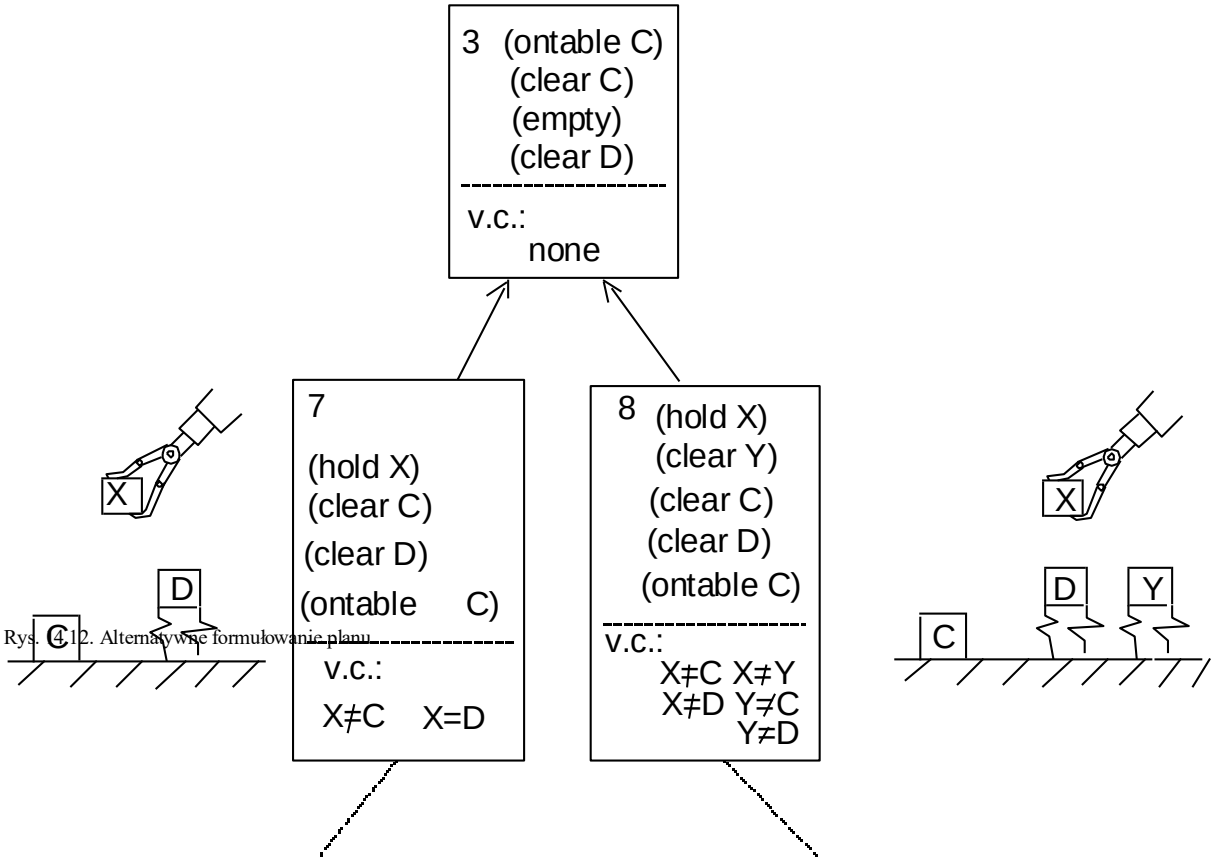




Rys. 14.10. Pierwszy i drugi krok w generowaniu planu

- cel (1): on C D
- szukamy O_i, który w add facts ma (on x y)
- jest nim: (puton x y)
- zastosowanie O_i generuje nowe fakty: (empty) i (clear x)
- w 2 - dwa nowe podcele: (hold C), (clear D)
- BPS szuka operatorów zawierających (hold x) lub (clear x) w części "add facts"
- podcel (hold C) jest spełniony przez (pickup C) i (takeoff C y)
- niespełnione podcele przenosimy do następnego bloku z warunkami wstępnymi
- w 4 konieczne sprawdzenie ograniczeń zmiennej y

Rozgałęzienia od bloku 3



Rys. 4.12. Alternatywne formułowanie planu

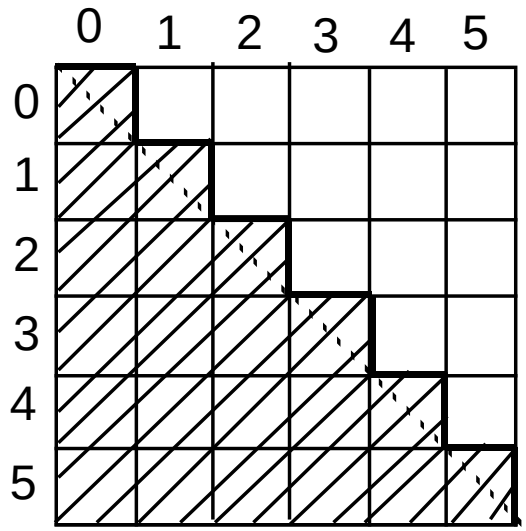
- od 3 i 4 – możliwe rozgałęzienie
- *means - end heuristic* (odległość od celu) –
- 3 jest bliższe (liczba wspólnych klauzul) stanowi wyjściowemu S_g niż 4
- wybieramy pójście ścieżkami z 3
- (putdown x) do bloku C (x=C) spełnia 3 spośród 4 klauzul - podceli z okna 3
- otrzymujemy sytuację jak w 2 - cykl
- operator (takeoff x C) dałby (on C C) - niedopuszczalny stan (sprzeczność)

Subplany – tabela trójkątna

- umożliwia zapamiętanie pewnych etapów planu, które mogą być wspólne dla różnych celów (S_{g1} - do pracy, S_{g2} - klubu, S_i - w domu)
- każdy z planów dla S_{g1} i S_{g2} ma wspólne sekwencje operatorów (pójść do auta, uruchomić silnik itd.)
- *Triangle Table* - mechanizm katalogowania planu i interakcji operatorów

0	Stan pocz. S_i	1: O_1		
1	Zawartość komórki powyższej minus *delete fact* operatora O_1	Dodane fakty *Add fact* O_1	2: O_2	
2	Zawartość komórki powyższej minus *delete fact* operatora O_2	Zawartość komórki powyższej minus *delete fact* operatora O_2	Dodane fakty *Add fact* O_2	3: O_3
				P: O_p
p				Dodane fakty *Add fact* O_p

Rozmiar tabeli trójkątnej



Liczba komórek w tabeli trójkątnej.

- Dla planu z p operatorów O_1, O_2,...,O_p, tablica składa się z:
- p+1 kolumn numerowanych od 0 do p
- p+1 wierszy, od 0 do p
- Liczba komórek w tabeli wynosi:

$$k = \frac{(p+1)^2}{2} + \frac{p+1}{2} = p \cdot \frac{p+3}{2} + 1$$

- dla p=4 będzie to 4*7/2+1=15 komórek, p=5: 21 komórek

- komórka może być pusta lub zawierać podzbiór klauzul opisujących stan systemu
- komórka (0,0) zawiera opis stanu początkowego S_i
- ostatni wiersz zawiera opis stanu docelowego S_g

Przykład tabeli zawierającej plan dla świata klocków

0	on BD, ontable C ontable D, clear A clear B, empty	1 takeof AC							
1	clear B, on BD ontable C, ontable D	hold A clear C	2 putdown A						
2	on BD, clear B, ontable C,ontable D	clear C	clear A, empty ontable A	3 pickup C					
3	on BD, clear B, ontable D	_____	clear A ontable A	hold C	4 puton CA				
4	on BD, clear B ontable D	_____	ontable A	_____	clear C, empty, on CA	5 takeoff BD			
5	ontable D	_____	ontable A	_____	clear C on CA	clear D hold B	6 putdown B		
6	ontable D	_____	ontable A	_____	clear C on CA	clear D	empty, clear B ontable B	7 takeoff CA	
7	ontable D	_____	ontable A	_____	_____	clear D	clear B ontable B	hold C clear A	8 puton CD
8	ontable D	_____	ontable A	_____	_____	_____	clear B ontable B	clear A	empty on CD clear C

Literatura do dalszego samodzielnego studiowania

Np.:

PLANNING ALGORITHMS

Steven M. LaValle

University of Illinois

Available for downloading at

http://planning.cs.uiuc.edu/

Published by Cambridge University Press

Następny wykład:

Przeszukiwanie
(Search problems)

http://www.lookhong.com/self/searching/