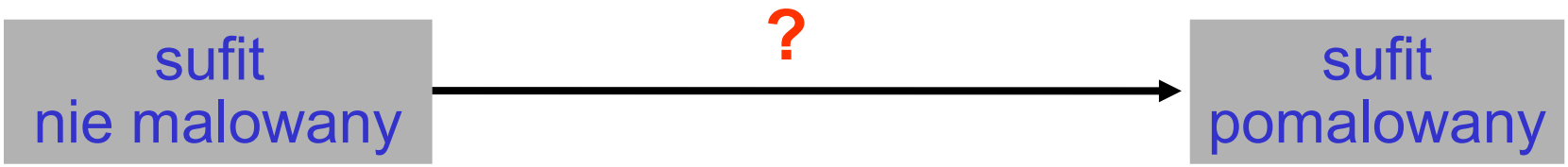
Automatyczne planowanie oparte na sprawdzaniu spełnialności

Linh Anh Nguyen
Instytut Informatyki
Uniwersytet Warszawski

Problem planowania



Złożoność

- Problem planowania jest NP-trudny
- Rozwiązania stosowane w praktyce:
  - Heurystyki
  - Metody działające skutecznie dla wystarczająco obszernej klasy zastosowań
  - Metody wyspecjalizowane dla konkretnego zastosowania.

Język STRIPS: **stany**

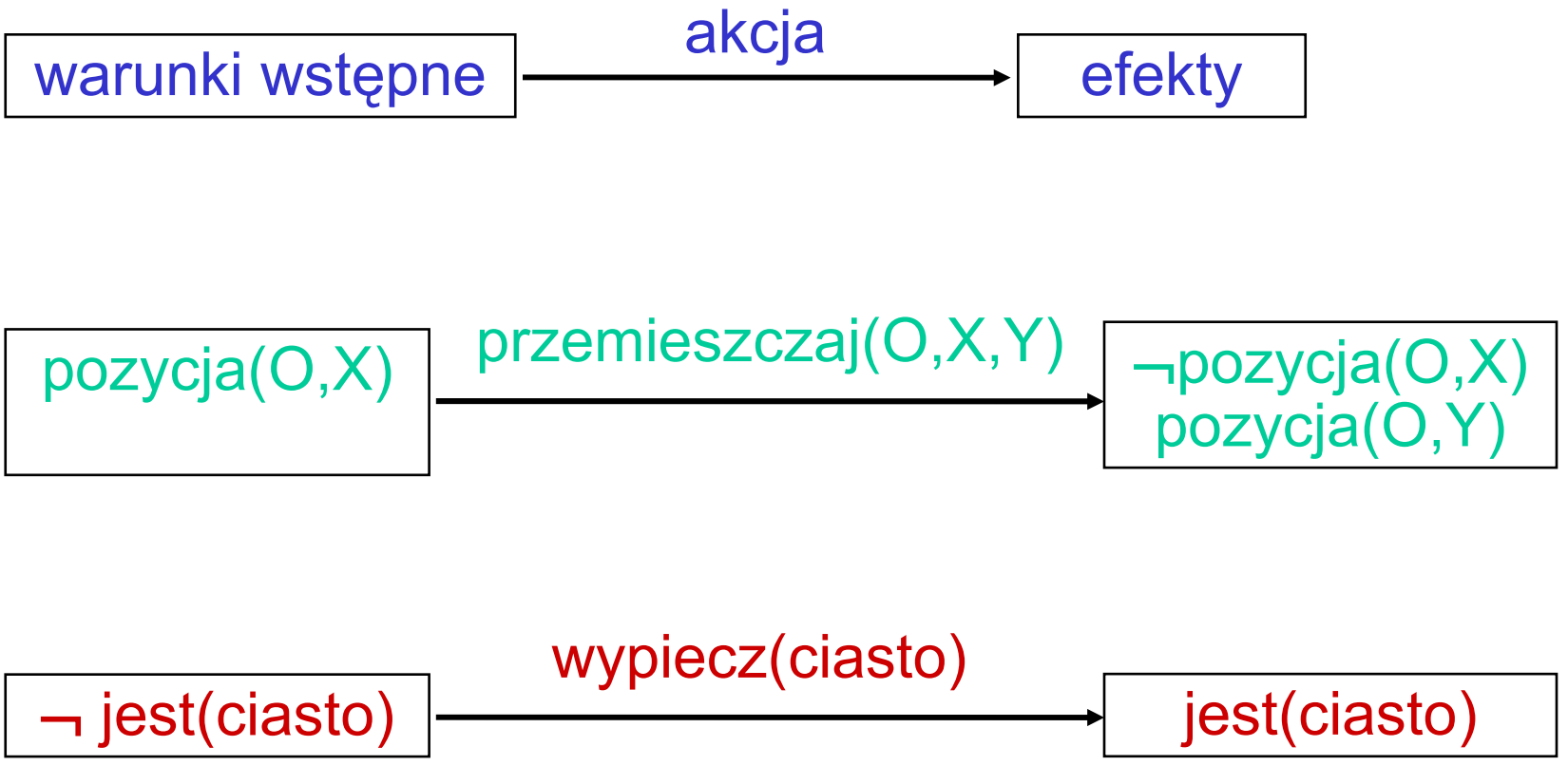
- Stan: zbiór literałów (formuł atomowych lub ich negacji)
- Przykład:

  pozycja(robot$_1$, pokój$_1$), pozycja(robot$_2$, pokój$_2$)

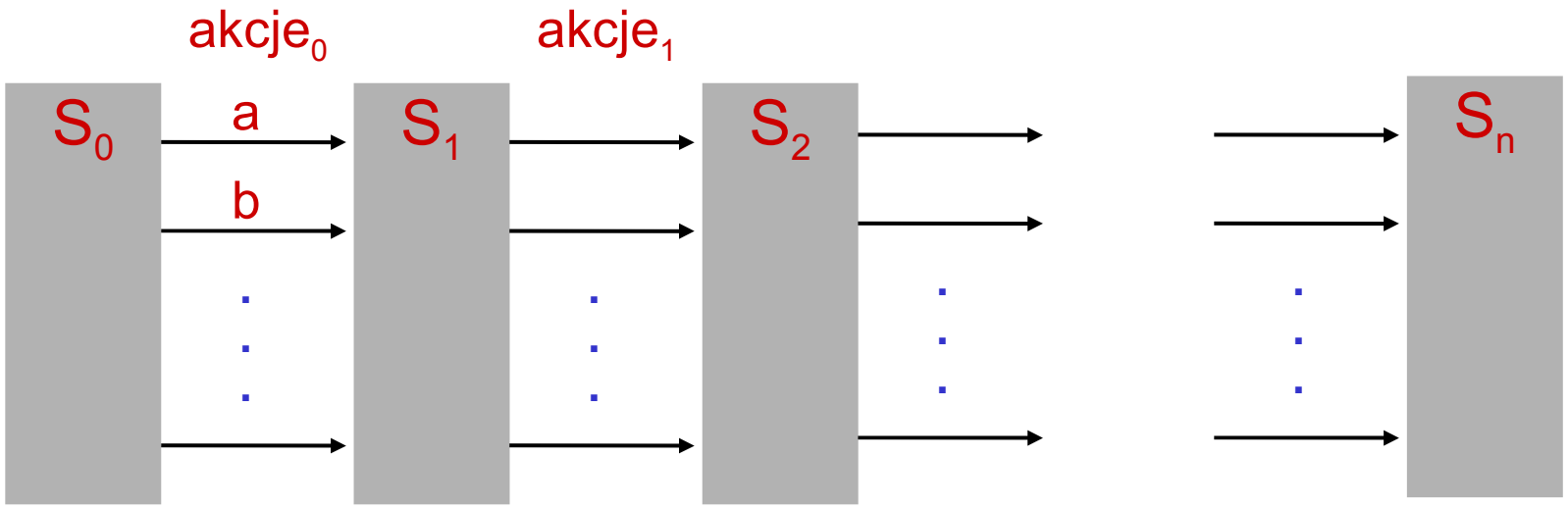
- Niedopuszczalne:

  pozycja(X,Y), pozycja(ojciec(jacek), pokój$_1$)

Język STRIPS: **akcje**



Język STRIPS: **postanowienie problemu**



Kiedy plan jest poprawny?

SatPlan: ogólne podejście

- Ograniczony problem planowania (P,n):
  - P jest problemem planowania, a n jest liczbą naturalną
  - każdy plan dla P o długości n jest rozwiązaniem dla (P,n)

- Algorytm planowania SatPlan:

  Wykonuj przeszukiwanie iteracyjnie pogłębiane:

  Dla n = 0, 1, 2, ...
  - Zakoduj (P,n) jako problem spełnialności zbioru formuł Φ
  - Jeśli Φ jest spełnialny, to wyekstrahuj plan dla P z wartościowania spełniającego Φ.

Przykład

Język:

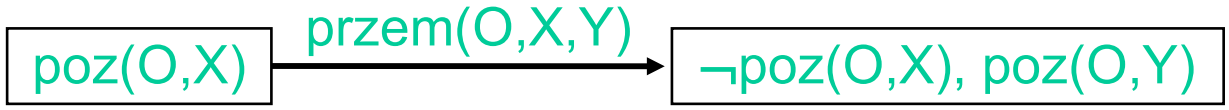
- robot r
- sąsiednie pokoje p_1, p_2
- akcja przemieszczania robota

Przykład: Zakodowanie (P,n) dla n=1

- Stan początkowy: {poz(r,p_1)}
  - Zakodowanie: poz(r,p_1,0) $\wedge$ $\neg$poz(r,p_2,0)
  - Ignoruj: $\neg$poz(r,r,0), $\neg$poz(l_1,l_2,0)
- Cel: {poz(r,p_2)}
  - Zakodowanie: poz(r,p_2,1)

Przykład: Zakodowanie (P,n) dla n=1

- Akcja:



- Zakodowanie:

$przem(r,p_1,p_2,0) \Rightarrow poz(r,p_1,0) \wedge poz(r,p_2,1) \wedge \neg poz(r,p_1,1)$

$przem(r,p_2,p_1,0) \Rightarrow poz(r,p_2,0) \wedge poz(r,p_1,1) \wedge \neg poz(r,p_2,1)$

- Ignoruj np. :

$przem(r,p_1,p_1,0) \Rightarrow poz(r,p_1,0) \wedge poz(r,p_1,1) \wedge \neg poz(r,p_1,1)$

$przem(p_1, \_, \_, 0) \Rightarrow ...$

$przem(\_, r, \_, 0) \Rightarrow ...$

Przykład: Zakodowanie (P,n) dla n=1

- Wzajemne wykluczanie akcji:

  $\neg przem(r,p_1,p_2,0) \vee \neg przem(r,p_2,p_1,0)$

- Aksjomaty tła

  (zmiany są wyłącznie wynikiem wykonania akcji):

  $\neg poz(r,p_1,0) \wedge poz(r,p_1,1) \Rightarrow przem(r,p_2,p_1,0)$

  $\neg poz(r,p_2,0) \wedge poz(r,p_2,1) \Rightarrow przem(r,p_1,p_2,0)$

  $poz(r,p_1,0) \wedge \neg poz(r,p_1,1) \Rightarrow przem(r,p_1,p_2,0)$

  $poz(r,p_2,0) \wedge \neg poz(r,p_2,1) \Rightarrow przem(r,p_2,p_1,0)$

Wyekstrahowanie planu

- Załóżmy, że
  - (P,n) jest zakodowany przez zbiór formuł Φ
  - Φ jest spełniony przy użyciu wartościowania V

- Wtedy krok i-ty planu dla (P,n) zawiera akcje a_i takie, że

$$V(a_i) = \text{true}$$

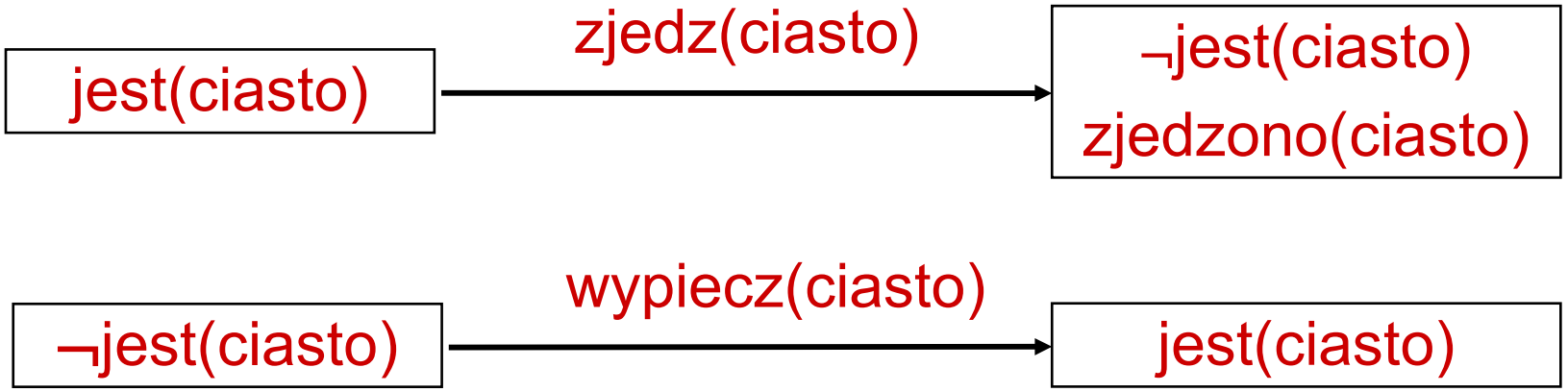
co oznacza, że została wybrana akcja a_i.

Zaawansowany algorytm SatPlan: idea

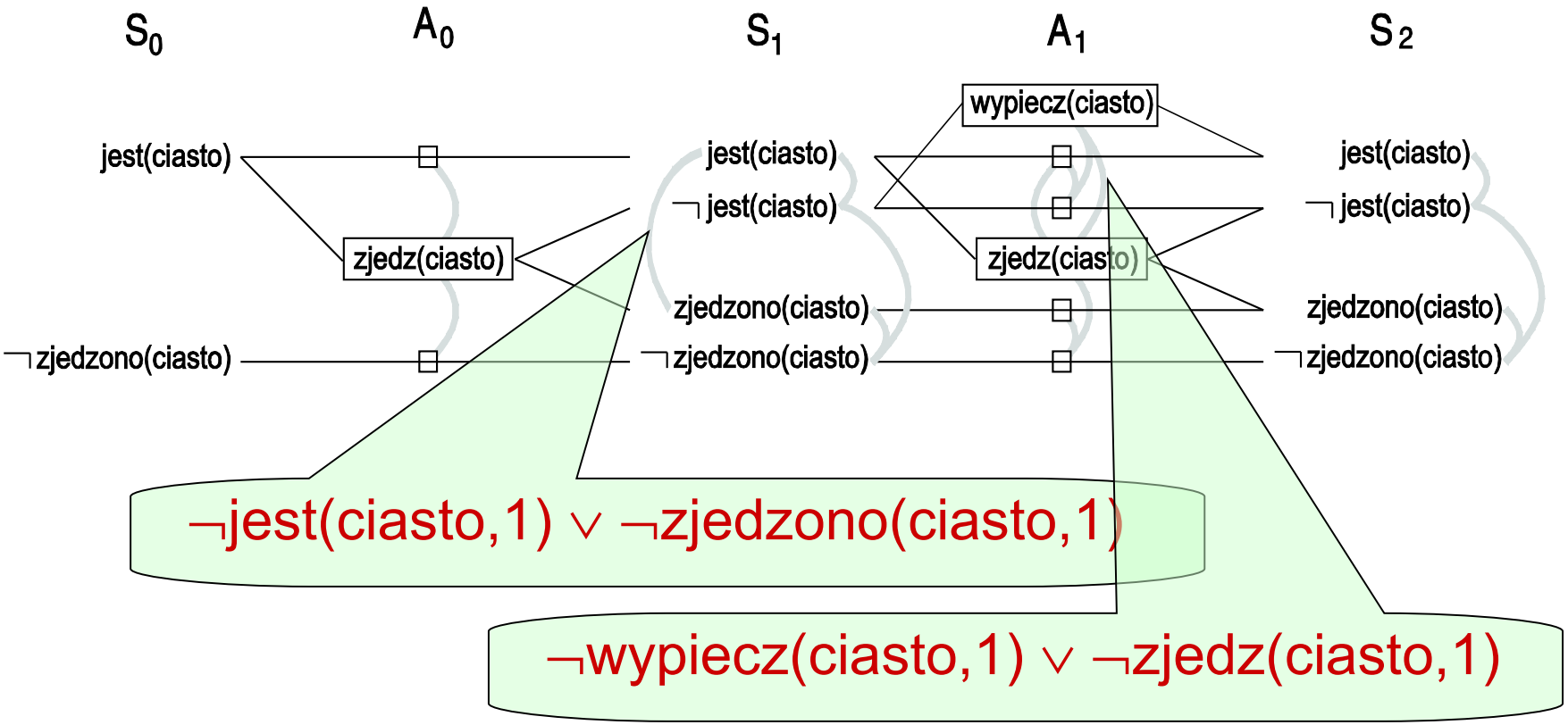
- Wykorzystać grafy planowania do ograniczenia przestrzeni przeszukiwania:
  - Sprawdzić warunek konieczny na istnienie rozwiązania dla (P,n).
  - Dodać do zakodowania dla (P,n) formuły wyrażające pewne więzy.

Graf planowania: przykład

- Stan początkowy: {jest(ciasto)}
- Cel: {jest(ciasto), zjedzono(ciasto)}
- Akcje:



Graf planowania: przykład



Zaawansowany algorytm SatPlan

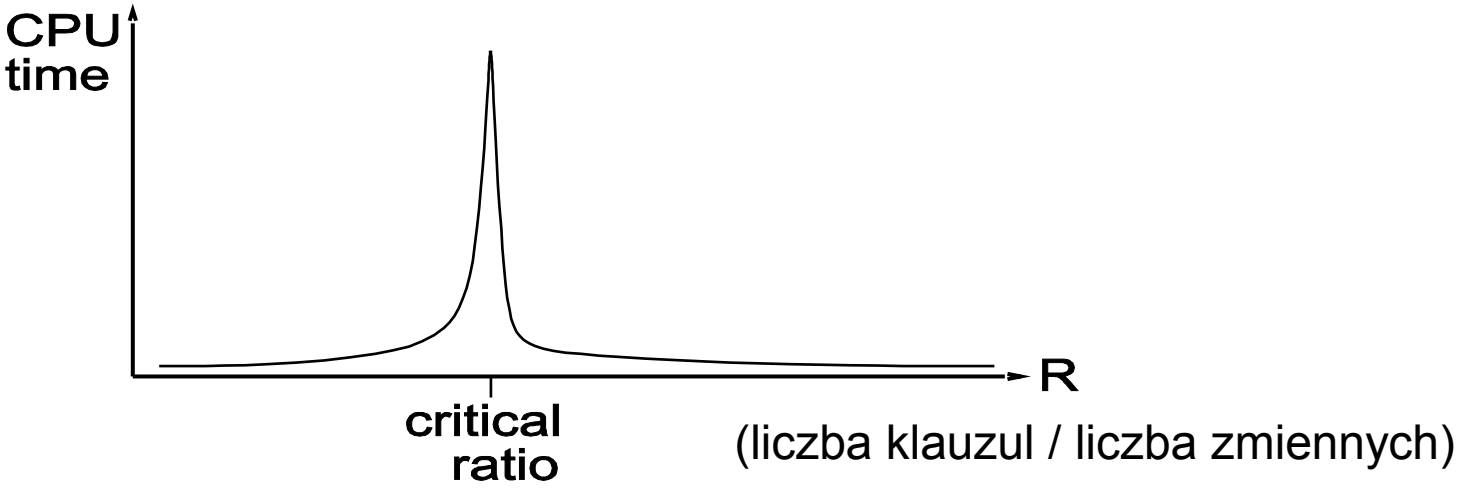
- Niech P będzie zadanym problemem planowania
- Dla n = 0, 1, 2, ...
  - Skonstruuj graf planowania dla (P,n)
  - Jeśli żadna para literałów celu nie jest w relacji wzajemnego wykluczania na ostatnim poziomie grafu, to:
    - Zakoduj (P,n) jako problem spełnialności zbioru formuł Φ uwzględniając relację wzajemnego wykluczania.
    - Jeśli Φ jest spełnialny, to wyekstrahuj plan dla P z wartościowania spełniającego Φ.

Własności algorytmu SatPlan

- SatPlan ukonkretnia akcje i literały.
- Ma przewagę kiedy:
  - dziedzina problemu nie jest za duża
  - generowany plan ma być (prawie) optymalny
  - trudność problemu nie jest związana
    - z analizą osiągalności w grafach planowania,
    - lecz z wyekstrahowaniem planu z grafów planowania

Skuteczność algorytmu SatPlan

- Programy sprawdzające spełnialność działają efektywnie dla większości praktycznych problemów.
- Są stale ulepszane i coraz efektywniejsze.



Skuteczność algorytmu SatPlan

- Implementacja algorytmu SatPlan (Kautz & Selman) zdobyła w kategorii optymalnego planowania na konkursach IPC (International Planning Competition):
  - pierwszą pozycję w IPC-2004
  - pierwszą pozycję w IPC-2006 (razem z innym prog.)
- Algorytm SatPlan jest jednym z wiodących podejść do automatycznego planowania.

Podsumowanie

- Planowanie jest jednym z najważniejszych zadań w systemach inteligentnych, w tym autonomicznych.
- Wysoka złożoność (NP-trudność) wymaga stosowania rozwiązań skutecznych dla wybranej dziedziny zastosowań.

Podsumowanie

- Sprowadzanie planowania do spełnialności rachunku zdań jest skutecznym zastosowaniem logiki:
  - Działa dla obszernej klasy zastosowań
  - Pozwala na bezpośrednią integrację z innymi rozwiązaniami opartymi na logice (np. dedukcyjnymi bazami danych, metodami reprezentacji wiedzy).