UNIVERSITÀ DEGLI STUDI DI BRESCIA
FACOLTÀ DI INGEGNERIA
DIPARTIMENTO DI ELETTRONICA PER L'AUTOMAZIONE

Generazione di Piani attraverso Grafi di Pianificazione

Corso di Intelligenza Artificiale

Alfonso E. Gerevini

Planning graph [Blum & Furst '95]

Generazione di piani in due fasi:

Costruzione "planning graph" G
+
Ricerca di un particolare sottografo di G

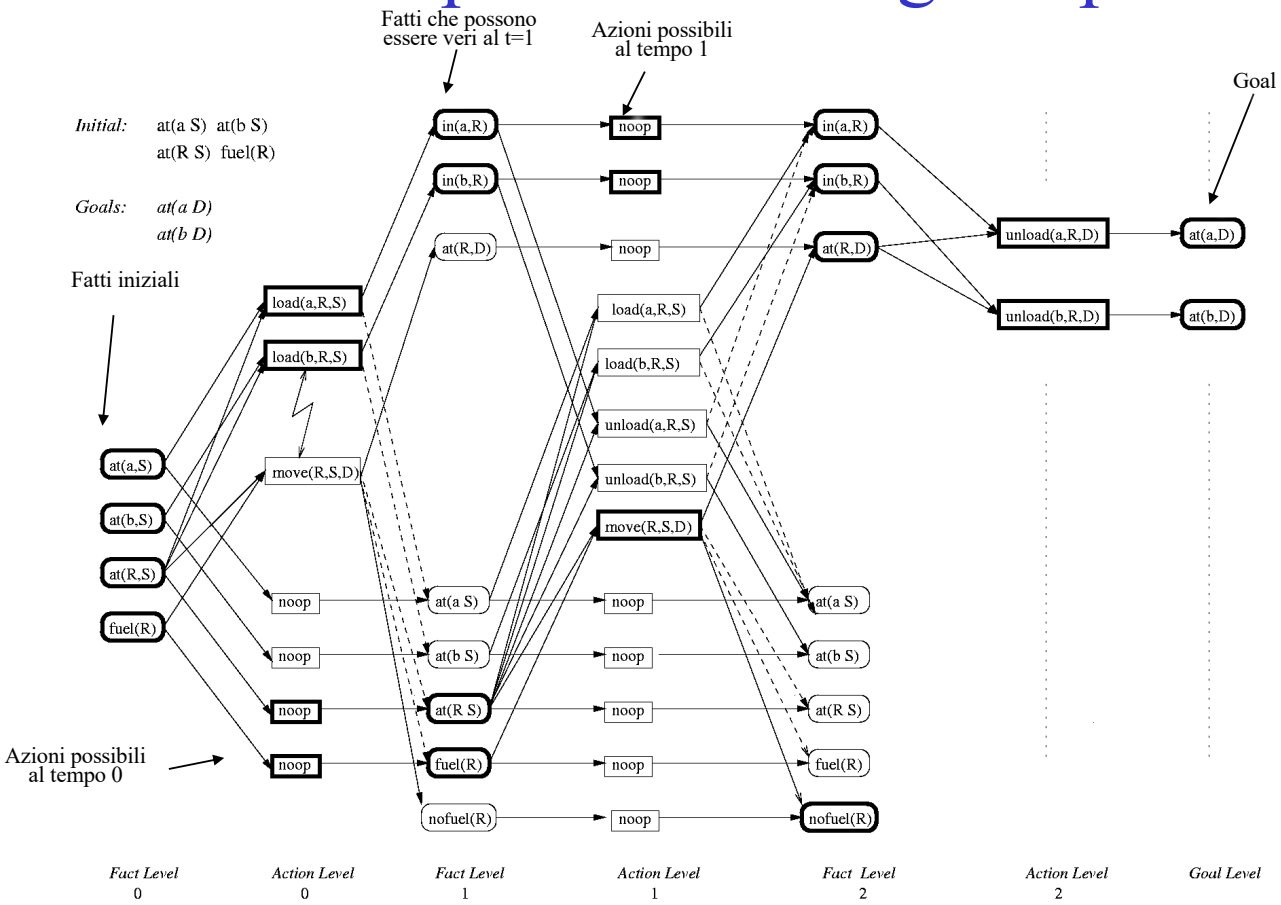
Planning graph

- Grafo aciclico diretto a livelli
- Livello: corrisponde ad un istante temporale ed è costituito da (sotto)livello dei fatti ($N_F(t)$) e (sotto)livello delle azioni ($N_A(t)$) (dove t indica l'istante temporale preso in considerazione).

- Nodi: fact-nodes → associati a proposizioni
  action nodes → associati ad azioni

- Archi: connettono action-nodes alle precondizione ed effetti
  - Archi precondizione
  - Archi Additivi
  - Archi Cancellanti

Esempio di Planning Graph



RELAZIONI DI MUTUA ESCLUSIONE:

–tra coppie di azioni (non possono essere pianificate allo stesso livello)

–tra coppie di proposizioni (non possono essere vere allo stesso livello)

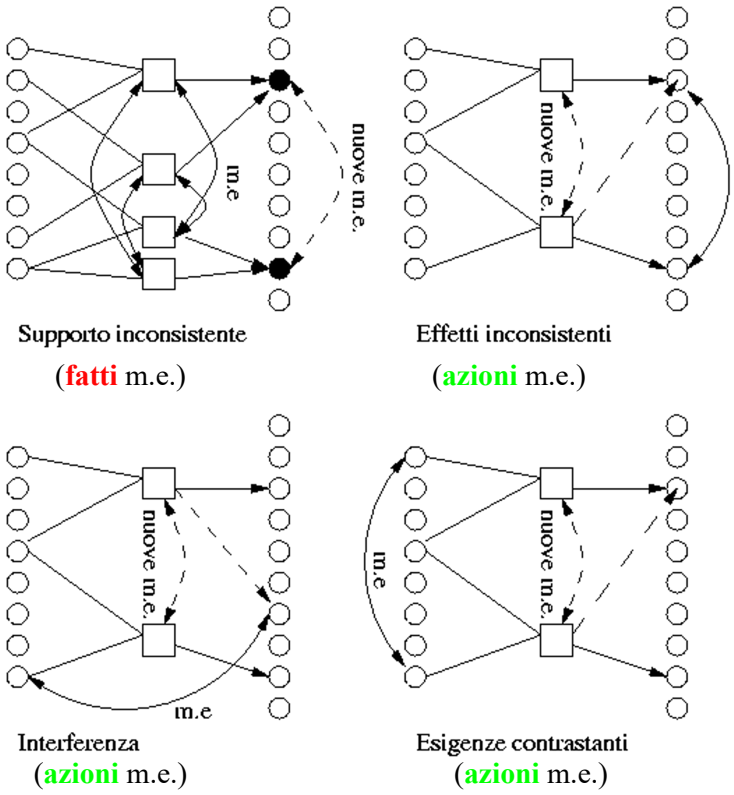
• Due azioni a_1 e a_2 sono *mutuamente esclusive* se:

– Interferiscono, cioè una cancella una precondizione o effetto additivo dell'altra

– Esigenze contrastanti, $\exists\ f_1 \in \varphi_0(a_1)$ e $f_2 \in \varphi_0(a_2)$ che sono mutuamente esclusivi

• Due fatti sono *mutuamente esclusivi* se $\neg\exists$ coppia di azioni non mutuamente esclusive che possa renderli veri allo stesso istante temporale

RELAZIONI DI MUTUA ESCLUSIONE:



Planning graph

Action Subgraph A è un sottografo del Planning Graph tale che se un nodo azione $a \in A$ allora:

- tutti i nodi/archi precondizione di *a* appartengono ad A
- tutti nodi di effetti additivi e gli archi additivi di *a* sono in A

Solution Subgraph S è un action subgraph tale che:

- tutti i nodi goal sono *supportati*
- ogni nodo-precondizione di ogni nodo azione è *supportato*
- non esistono copie di nodi azione mutuamente esclusivi

Un fact-node *q* al livello *i* in *S* è *supportato* se

- al livello *i-1* di S esiste un nodo azione connesso a *q* da un arco additivo, o
- *q* è un fact-node del livello iniziale (*i*=0).

Blocks World (4 operators)

```
(:action pick-up
        :parameters (?ob1)
        :precondition (and (clear ?ob1) (on-table ?ob1) (arm-empty))
        :effect (and (not (on-table ?ob1)) (not (clear ?ob1))
                (not (arm-empty)) (holding ?ob1)))

(:action put-down
        :parameters (?ob)
        :precondition (holding ?ob)
        :effect  (and (not (holding ?ob)) (clear ?ob) (arm-empty) (on-table ?ob)))
```

Blocks World (4 operators)

(:action stack

:parameters (?sob ?sunderob)

:precondition (and (holding ?sob) (clear ?sunderob))

:effect (and (not (holding ?sob)) (not (clear ?sunderob)) (clear ?sob)
(arm-empty) (on ?sob ?sunderob)))

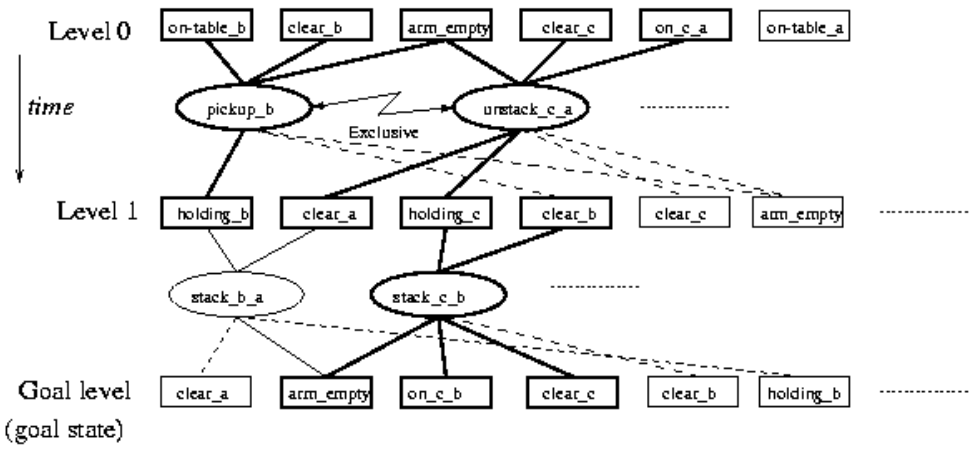
(:action unstack

:parameters (?sob ?sunderob)

:precondition (and (on ?sob ?sunderob) (clear ?sob) (arm-empty))

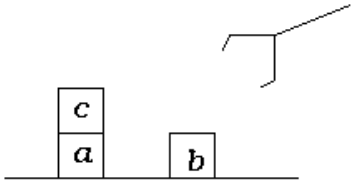
:effect (and (holding ?sob) (clear ?sunderob) (not (clear ?sob))
(not (arm-empty)) (not (on ?sob ?sunderob)))))

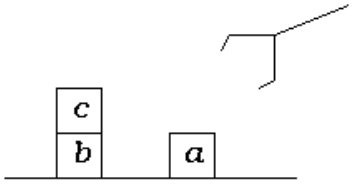
Esempio di Action Subgraph



Initial: on-table(a) on-table(b) on(c a)
clear(c) clear(b) arm-empty()

Goal: clear_a arm_empty
on_c_b clear_c





Costruzione del Planning graph

- t=0; $N_F(0)$= Fatti iniziali; $N_A(0)= \varnothing$
- ***Repeat***
  1. Aggiungi a $N_A(t)$ tutte le azioni le cui precondizioni appartengono a $N_F(t)$ e non sono mutuamente esclusive;
  2. Aggiungi a $N_A(t)$ un'azione no-op per ogni fatto di $N_F(t)$;
  3. Aggiungi a $N_F(t+1)$ tutti gli effetti additivi delle azioni di $N_A(t)$;
  4. Determina e memorizza quali azioni di $N_A(t)$ sono mutuamente esclusive;
  5. Determina e memorizza quali fatti di $N_F(t+1)$ sono mutuamente esclusivi;
  6. $t := t+1$;
- ***Until***

  (A) $N_F(t+1)$ contiene tutti i goal e questi non sono M.E; **oppure**

  (B) il grafo è "*stabilizzato*"
- *if vale* (A) *then* avvia la ricerca al livello t;

  else (B) *return FAIL*;

Planning graph

- I livelli di un Planning Graph *crescono monotonamente*:
  - se la proposizione "$f \in$ livello t "$\Rightarrow$ "$f \in$ livello t +1"
  - se p e q non sono M.E. al livello t non lo saranno al livello t+1
- Il tempo e lo spazio necessari per costruire un Planning Graph sono polinomiali rispetto ai parametri di ingresso (numero fatti e azioni)
- Un planning graph è stabilizzato al tempo t se $|N_F(t)|= |N_F(t+1)|$
- Un problema di pianificazione **non** ha soluzione se il suo planning graph è stabilizzato al livello t_s e:
  - un goal $g \notin N_F(t_s)$, oppure
  - 2 goal sono mutuamente esclusivi in $N_F(t_s)$

Cond suff. ma non necessaria

Fase di ricerca: Algoritmo Searchplan

Input: *Insieme* di (sotto)goal ***goal***, livello corrente *t*;

Output: *TRUE* se esiste *insieme* di azioni che realizzano **goal**, *FALSE* altrimenti

1. if (t=0) then return *TRUE*;
2. if (**goal** è stato "memoizzato" in t) return FALSE;
3. Scegli un insieme A di azioni non M.E. che supportano **goal** (*PUNTO DI BACKTRACKING*)
   - **subgoals**:=precondizioni delle azioni in A;
   - if (**Searchplan(subgoals, t-1)**=TRUE) return TRUE;
4. "Memoizzazione" di **goal** al livello t;
5. return FALSE

Memoizzazione e Test di Terminazione

Memoizzazione di **subgoals**

Memorizzo i fatti in **subgoals** al livello *t* in una *hash table H*.

Se la ricerca genera un insieme di subgoals "che sono in H", allora restituisce *FALSE*;

To Memoize: "To store (the result of a computed expression) so that it can be subsequently retrieved without repeating the computation"

Test di terminazione (per algoritmo GraphPlan - vedi slide)

Se il **grafo è stabilizzato** ad un livello **n** e l'insieme di insiemi di subgoals memoizzati per il livello **n** dalla ricerca corrente è uguale a quello della ricerca precedente, allora l'algoritmo si interrompe restituendo FALSE ("no plan exists"). Dimostrazione nell'articolo di Blum & Furst (AIJ '95)

Worst case: complessità spaziale esponenziale, in pratica meno usando tecniche per ottimizzare H (ad es. memorizzo solo gli insiemi di goals **più piccoli**)

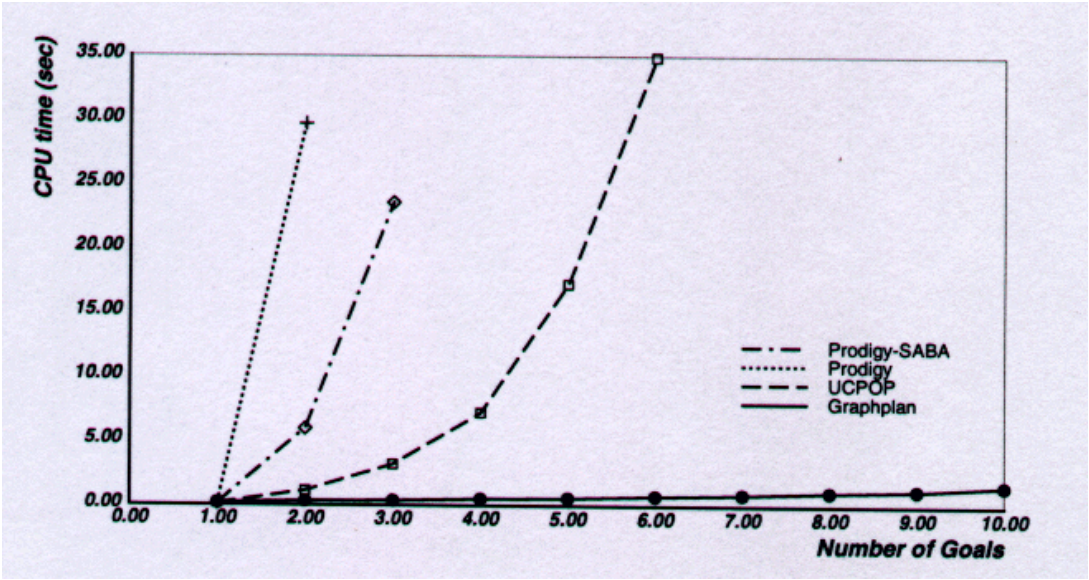
Algoritmo GraphPlan

Input: problema di pianificazione $\langle I, G, O \rangle$

Output: un *piano* se il problema ammette soluzione
FALSE altrimenti

1. Costruzione Planning Graph
2. t = livello finale di G
3. While(Searchplan(G, t)=FALSE)
   - espansione planning graph (fasi 1÷6)
   - if ("**test di terminazione**"=TRUE) return FALSE;
   - t:=t+1;
4. Return l'insieme di azioni selezionate ad ogni livello di G da Searchplan;

Risultati sperimentali (dominio Rocket)



Esercizio

Si consideri il seguente semplice problema nel mondo dei blocchi:

- Azioni:
  - "sposta blocco X da blocco Y a blocco Z"
  - "Sposta blocco X da blocco Y a tavolo (T)"
- Stato iniziale: on(A,B), on(B,C), on(C,T), Clear(A)
- Goal: on(A,T), on(B,T), on (C,T)

Costruire il planning graph del problema fino al livello in cui si stabilizza

Indicare tre possibili insiemi di sottogoals generati dalla procedura di ricerca