

Università degli Studi di Brescia
Facoltà di ingegneria
Dipartimento di Elettronica per l'Automazione

Generazione di Piani attraverso Grafì di Pianificazione

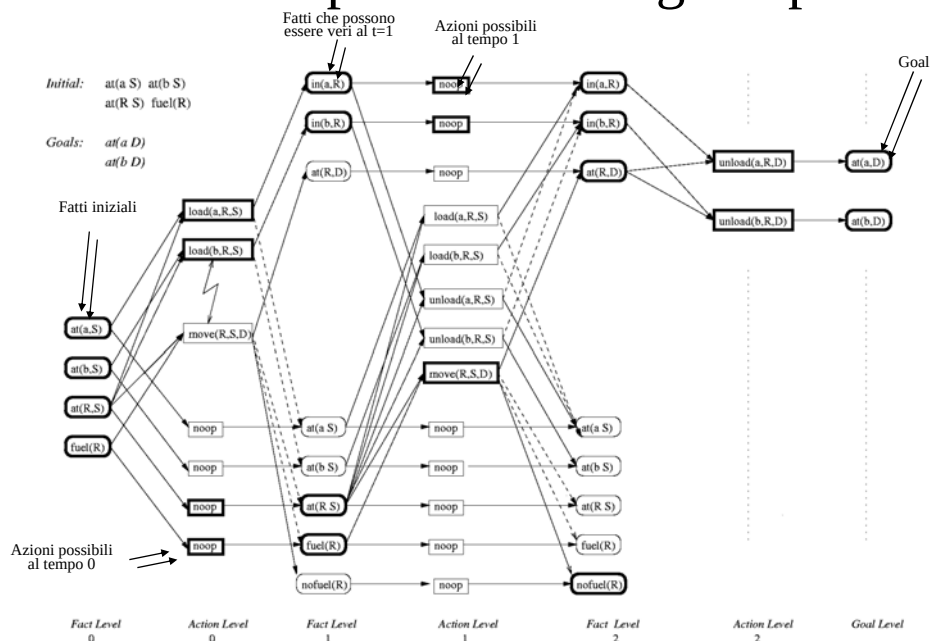
Planning graph [Blum & Furst '95]

Generazione di piani =
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 $\rightarrow$ associati a proposizioni
action nodes $\rightarrow$ associati ad azioni
- Archi: connettono action-nodes alle precondizioni ed effetti
 - Archi precondizione
 - Archi Additivi
 - Archi Cancellanti

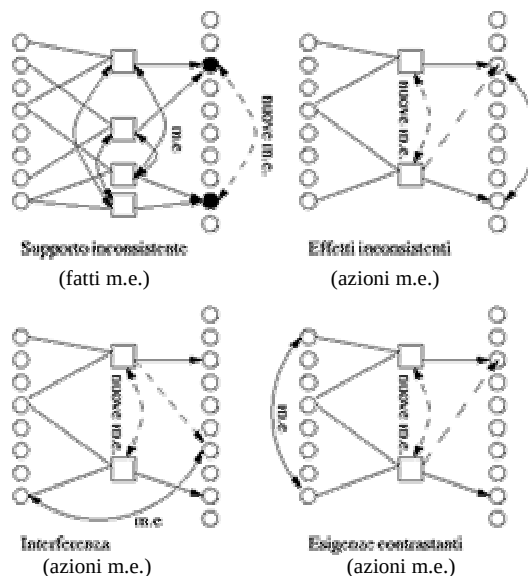
Esempio di Planning Graph



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 preconditione 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

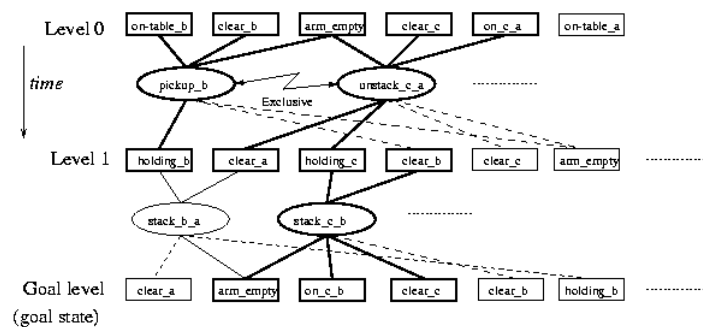
Solution Subgraph S è un action subgraph tale che:

- 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

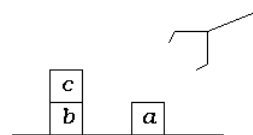
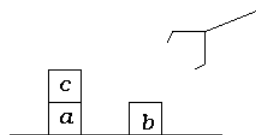
- al livello $i-1$ di S esiste un nodo azione connesso a q da un arco additivo, o
- q è un fact-node del livello iniziale ($i=0$).

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; $N_A(0)= \emptyset$
- **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 (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 “ $f \in \text{livello } t$ ” $\Rightarrow$ “ $f \in \text{livello } t+1$ ”
 - 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
- 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_1 e:
 - un goal $g \notin N_F(t_1)$, oppure
 - 2 goal sono mutuamente esclusivi in $N_F(t_1)$

Fase di ricerca (searchplan)

Input: Insieme di sottogoal **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”) 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**;

Test di terminazione

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)

