Problemi di pianificazione

Dato un agente in grado di compiere delle azioni nel mondo in cui è immerso, un **problema di pianificazione**, riguarda la scelta di un piano di azioni, che l'agente può compiere al fine di portare il mondo dal suo stato attuale allo stato di goal.

È possibile definire un problema di pianificazione come una quintupla:

$$< P, A, I, G, \tilde{G} >$$

dove:

- $P \neq \emptyset$ è un insieme finito di proposizioni;
- $A \neq \emptyset$ è un insieme finito di azioni, composto da una quintupla $< n, R, \tilde{R}, T, \tilde{T} >$:
  - n è un simbolo che identifica il nome dell'azione;
  - R è un sottoinsieme di P detto precondizione asserita;
  - $\tilde{R}$ è un sottoinsieme di P detto precondizione negata;
  - T è un sottoinsieme di P detto postcondizione asserita;
  - $\tilde{T}$ è un sottoinsieme di P detto postcondizione negata;
- $I \neq \emptyset$ è un sottoinsieme finito di P detto stato iniziale;
- $G \neq \emptyset$ è un sottoinsieme finito di P detto goal asserito;
- $\tilde{G} \neq \emptyset$ è un sottoinsieme finito di P detto goal negato;

In ogni istante, il mondo è caratterizzato da uno stato.

Per risolvere un problema di pianificazione è possibile operare una ricerca nello spazio degli stati che parta dallo stato iniziale I e proceda all'identificazione delle soluzioni al problema, dove i nodi vengono organizzati in un albero di ricerca.

Fissato un percorso nell'albero di ricerca, l'insieme dei nodi costruiti e non ancora scartati viene detto frangia.

Se ogni nodo viene:

- Aggiunto in fondo alla frangia, allora l'algoritmo di ricerca nello spazio degli stati viene detto in ampiezza (BFS Breadth-First Search), nel quale in un istante dobbiamo avere in memoria tutti i figli del nodo espanso.
- Aggiunto in fondo alla frangia, allora l'algoritmo di ricerca nello spazio degli stati viene detto in profondità (DFS Depth-First Search), nel quale la quantità di memoria da usare è lineare rispetto alle azioni necessarie per il goal.
- Aggiunto in modo tale che gli stati più vicini agli stati di goal siano più vicini alla testa della frangia, allora l'algoritmo di ricerca nello spazio degli stati viene detto informato (è una via di mezzo tra BFS e DFS).

Tutti i metodi hanno complessità computazionale asintotica di caso pessimo di esponenziale di ordine $O(b^d)$, dove $b = |A|$ e d è il numero di azioni necessarie per raggiungere, nel caso pessimo uno stato di goal. Viceversa,

1. BFS ha complessità computazionale spaziale asintotica di caso pessimo di ordine $O(b^d)$ e garantisce che per problemi risolubili, venga ottenuta una soluzione composta dal numero minimo di azioni.
2. DFS ha complessità computazionale spaziale asintotica di caso pessimo di ordine $O(d)$ pur non garantendo, che per problemi risolubili, venga ottenuta una soluzione composta dal numero minimo di azioni.

Per ottenere un buon compromesso tra BFS e DFS di solito si utilizza una ricerca ad approfondimenti successivi (ricerca informata). Nella quale l'obbiettivo è quello di tenere basso l'uso di memori, lo svantaggio è che tutte le volte viene costruito l'albero.

Normalmente, la ricerca informata, richiede che:

1. Il costo sia definito positivo (mai nullo o negativo) e che sia additivo.
2. Venga definita una funzione di costo per ogni nodo N, che quantifichi il costo minimo necessario per partire dalla radice, passare per il nodo N e raggiungere lo stato di goal.

La ricerca nello spazio degli stati può essere:

- in avanti (foward chaning) → parte dallo stato iniziale fino ad arrivare ad uno stato di goal.

- all’indietro (backward chaning) → parte dallo stato di goal fino ad arrivare allo stato iniziale, invertendo il ruolo delle postcondizioni con le precondizioni.