**CASTELLO DELLA MEMORIA: PROBLEMI DI RICERCA**

|  |  |
| --- | --- |
| **RICERCA**  **SU**  **GRAFO** | Agente necessita di una rappresentazione del problema per computare la soluzione. |
|  | La struttura più idonea risulta il GRAFO. |
|  | Il grafo è costituito da nodi i quali rappresentano gli stati |
|  | Stati rappresentano la descrizione in un dato momento dell’ambiente in cui l’agente si trova ad operare. |
|  | Una soluzione vista come un cammino per il grafo. |
|  | Funzione goal che valuta se uno stato è soluzione ovvero obiettivo del problema. |
|  | Frontiera idea alla base della ricerca. |
|  | Espansione della frontiera in modo incrementale per trovare la soluzione. |
|  | Ricerca non informata ovvero non si ha conoscenza della posizione dell’obiettivo. |
|  | BFS con coda FIFO |
|  | DFS con Pila LIFO |
|  | ID iterative deepening unione di BFS + DFS |
|  | LCFS coda prioritaria con funzione di costo ordinatrice |
|  | Ricerca Informata o ricerca euristica |
|  | Funzione euristica la quale stima il costo per giungere all’obiettivo |
|  | L’euristica deve essere ammissibile ovvero deve sempre sottostimare il costo reale per giungere alla soluzione |
|  | DFS con euristica comporta scelta del percorso che va a finire per primo nello stack |
|  | GBFS ricerca greedy |
|  | A\* search = LCFS + GBFS |
|  | Ammissibilità e completezza dell’A\* |
|  | IDA\* invece di usare la profondità il limite è dato dalla funzione di costo dell’A\* |
|  | CP cycle pruning pota I percorsi con un nodo doppione |
|  | MP multi-path pruning |
|  | Closed list per memorizzare i nodi già visitati |
|  | Depth First search branch & bound |
|  | Schiacciamento della soluzione tra il lower bound e l’upper bound |
|  | Ottimalità garantita tramite sequenza di soluzioni incrementale |
|  | Ricerca Bidirezionale |
|  | Foreward search |
|  | Backward search |
|  | Ricerca per isole |
|  | Scomposizione del problema in sottolivelli o sotto problemi |
|  | Ricerca con programmazione dinamica |