Costruzione del caso peggiore per un algoritmo deterministico

Un albero del gioco, $T_{3,6}$ è un albero uniforme i cui tutti i nodi interni hanno 3 figli che corrispondono alle mosse eseguibili dalla configurazione padre. L'albero in questione avrà la bellezza di $3^{2*6} = 9^6 =$ 531.441 foglie, quindi, richiederebbe parecchio tempo disegnarlo.

Il problema della valutazione del gioco consiste nel determinare il valore restituito dalla radice tenendo conto che ogni foglia ha associato un numero reale che corrisponde al valore del gioco e che ogni nodo se si trova a distanza dispari dalla radice sarà di tipo MAX e quindi restituirà il valore massimo dei suoi d figli, altrimenti sarà di tipo MIN e restituirà il valore minimo.

Nel caso in cui ogni foglia potesse assumere solo il valore 0 o 1 MAX potrà essere considerato equivalente ad un OR e MIN equivalente ad un AND.

Un esempio di albero per cui un algoritmo deterministico dovrebbe valutare tutte le foglie è il seguente:

Considerando una procedura deterministica ricorsiva nella quale l'albero viene letto da sinistra verso destra partiamo ricorsivamente dalla radice e scendiamo nei sottoalberi sinistri fino a raggiungere una foglia. A questo punto guardiamo il padre della foglia in questione:

- Caso OR:

Se si tratta di un OR per raggiungere il nostro obbiettivo bisogna fare il modo che non si possa stabilire il valore del padre senza aver prima analizzato gli altri due figli quindi sarà necessario che il figlio più a sinistra e quello al centro siano 0 ed a quel punto il figlio restante dovrà avere valore 1. Il figlio più a destra dovrà avere necessariamente il valore 1 per fare in modo che anche il padre assuma il valore 1 costringendo così l'algoritmo a dover leggere gli altri sottoalberi per dare un valore all'AND successivo. In questo modo l'algoritmo passerà ad analizzare il sottoalbero immediatamente a destra di quello appena analizzato e sarà necessario che anche questo abbia valore 1, ed essendo anche lui un OR dovrà avere le caratteristiche del sottoalbero visto prima.

Analizzato anche il secondo sottoalbero, il quale come già detto avrà valore 1, l'algoritmo non potrà accora dare un valore all'AND poiché questo dipenderà dal valore del terzo e ultimo sottoalbero che per nostro interesse dovrà avere valore 0 in modo da rendere l'AND di valore 0 ed obbligare l'algoritmo a dover analizzare i restanti sottoalberi per poter dare un valore all'OR successivo in modo analogo ad il primo OR analizzato e così via fino alla radice.

- Caso AND:

Se invece si tratta di un AND bisogna fare in modo che l'algoritmo non possa escludere prima di avere analizzato gli altri due figli se quell'AND assuma valore 0 o 1, quindi i due figli dovranno avere valore 1 e il restante dovrà assumere il valore 0 rendendo così il padre di valore 0 e costringendo l'algoritmo a dover analizzare gli altri due sottoalberi per poter assegnare un valore all'OR seguente.

Analizzando il secondo sottoalbero per dare un valore all'OR sarà necessario che anche questo abbia valore 0 e quindi dovrà essere uguale al sottoalbero più a sinistra. In questo modo l'algoritmo non potrà assegnare un valore all'OR fino a quando non avrà analizzato anche il restante sottoalbero, il quale dovrà assumere valore 1 in modo tale da rendere l'AND successivo dipendente dall'ultimo suo figlio e così via fino alla radice.

Quindi ricapitolando: Bisogna costringere l'algoritmo a dover analizzare tutto l'albero, perciò, bisogna che analizzando un OR i due figli più a sinistra abbiano valore 0 in modo tale da rendere l'OR dipendente dall'ultimo figlio (quello più a destra), il quale dovrà assumere valore 0 o 1 a seconda che si tratti di uno dei due figli più a sinistra o il figlio più a destra di un AND.

Analizzando un AND bisogna fare in modo che l'algoritmo non possa escludere senza prima aver analizzato tutti e tre i figli che quell'AND possa assumere il valore 1 perciò i due figli più a sinistra varranno 1 e l'ultimo, a seconda se ci troviamo nei due figli più a sinistra o in quello più a destra di un OR, varrà o 0 o 1.