Costruzione del caso peggiore per un algoritmo deterministico

Fatte le necessarie assunzioni, produci un esempio di un albero di un gioco T3,6 per il quale un algoritmo deterministico deve valutare tutte le foglie. Non disegnare l'albero (perché?) ma spiega come deve essere costruito.

Si prende in considerazione un gioco a due giocatori in cui le mosse vengono effettuate a turno e se uno dei due giocatori vince, l'altro perde, perciò non esiste una situazione di pareggio. Quindi si produce un albero di gioco, dove gli archi rappresentano le mosse e nodi le diverse configurazioni del gioco.

Per costruire l'albero di gioco T3,6:

d=3 che che corrisponde ai figli, ovvero mosse che vengono eseguite dalla configurazione padre.

$d^{2k}=3^{12}=531441$ foglie e si trovano a distanza 2k=12.

il problema della valutazione del gioco consiste nel determinare il valore restituito dalla radice tenendo conto che ogni foglia ha associato un numero reale e che ogni nodo restituisce il valore massimo associato ai suoi d figli.
I nodi interni a distanza pari dalla radice sono di tipo MIN, a distanza dispari di tipo MAX.

Ogni foglia ha valore reale, per facilità si assegna 1 o 0, se il nodo è di tipo MAX allora il valore sarà dato da OR, se il nodo è di tipo MIN allora il valore sarà dato da AND.

Per poter valutare in modo efficiente tutte le foglie dell'albero T3,6 si dovrà analizzare l'albero di gioco a ritroso.
All'ultimo livello, quindi ai nodi finali il giocatore, chiamiamo Alice dovrà attuare il suo turno di gioco il cui obbiettivo è quello di minimizzare. Alice valuta i valori minimi dei tre nodi e li assegna al nodo padre, questo in modo da minimizzare la sua perdita.
Al livello superiore, perciò il penultimo dovrà agire il giocatore che avrà l'obiettivo di massimizzare il valore, che chiameremo Carlo. Carlo attuerà lo stesso procedimento di Alice ma agendo sui nodi con valori maggiori.
Questo procedimento alternato, continuerà a ritroso fino al primo livello dove toccherà nuovamente ad Alice che sceglierà tra i nodi il valori minore e lo porterà al nodo padre, in questo caso la radice.

Dato l'albero T3,6 è anche possibile esaminarlo senza valutare tutti i nodi, ignorando alcuni rami, facendo riferimento a due valori che rappresentano MIN e MAX ne vengono aggiornati man mano che si procede.

In questo si può definire che se si ha un buon ordinamento il numero di nodi esaminati diviene $O(\sqrt{d^{2k}}) = O(\sqrt{3^{12}})$.

Nel caso peggiore bisogna fare in modo che venga considerato ogni nodo dell'albero in modo ricorsivo.
Se ci troviamo nel caso dell'AND si dovrà fare in modo che al padre venga assegnato il valore 0, così da rendere necessaria l'analisi dei nodi nei seguenti sottoalberi per assegnare un valore nel caso OR. Ritrovandosi nel caso in cui il valore sarà dato da OR i primi due figli da sinistra dovranno avere il valore zero in modo che l'algoritmo debba visitare anche il terzo figlio d per poter valutare quale valore assegnare al padre, perciò il figli più a destra dovrà avere necessariamente il valore uno, questo farà sii che nel successivo caso AND l'algoritmo debba per forza valutare i sottoalberi per determinare il valore. Il sottoalbero successivo dovrà essere 1 in modo che sia necessariao prendere in considerazione il prossimo sottoalbero che richiederà di rendere l'AND uguale a zero, in modo che debbano essere valutati tutti i sottoalberi per poter definire la radice.

Nel caso peggiore sarà dato dalla considerazione di ogni nodo dell'albero e perciò dato semplicemente dal numero di figli per ogni nodo e dalla profondità di ricerca perciò $O(d^{2k}) = O(3^{12})$.