Analisi di complessità dell’algoritmo

L’algoritmo è diviso in due fasi: una fase di inizializzazione nella quale i lotti vengono ordinati secondo un criterio di priorità e numero di lavorazioni, ed una seconda dove avviene la schedulazione. Si definisca ***n*** la dimensione dell’input dell’algoritmo, ovvero il numero di lotti da schedulare

Analisi prima fase: la prima fase è caratterizzata principalmente da un ordinamento. Utilizzando l’algoritmo MergeSort la complessità è theta(n\*lgn) poiché esegue sempre tutti i confronti strettamente necessari (si dimostra questi essere omega(n\*lgn)).

Analisi seconda fase: il primo foreach esegue theta(n) operazioni poiché visita una ed una sola vota ogni lotto nella lista. Il ciclo while esegue O(maxSlot) operazioni nel caso peggiore (quando non si riesce a schedulare una lavorazione in nessuno slot) e 1 sola operazione nel caso migliore (quando il lotto ha un’unica lavorazione ed il primo slot per schedularla è libero). Il secondo foreach ha complessità O(size(macchine)) nel caso peggiore (non si trova nessuna macchina su cui schedulare quella lavorazione in quello slot) mentre complessità O(1) nel caso migliore (quando la prima macchina nella lista può prendere in carico la lavorazione). E’ interessante valutare la complessità totale della seconda parte dell’algoritmo nel caso peggiore, la quale è theta(n)\*O(maxSlot)\*O(size(Macchine)). Poiché gli ultimi due termini sono costanti (non dipendono dall’input) possono essere trascurati e la complessità nel caso peggiore (ed anche nel migliore) è theta(n).

Poiché nella seconda parte la complessità è theta(n), questa viene dominata dalla prima parte, e l’algoritmo ha complessità totale theta(n\*logn), quindi sia nel caso migliore che nel caso peggiore. Immagine che contiene testo

Descrizione generata automaticamente