Unione europea
Fondo sociale europeo

MINISTERO DEL LAVORO
E DELLE POLITICHE SOCIALI
Direzione Generale per le Politiche
per l'Orientamento e la Formazione

REGIONE DEL VENETO

Schedulazione di attività in presenza di attività interrompibili

Maria Silvia Pini

Resp. accademico: Prof.ssa Francesca Rossi

Università di Padova

Attività FSE DGR 1102/2010

La gestione dell'informazione nell'azienda

Schedulazione intelligente di attività in presenza di risorse limitate e matching stabile ed efficiente tra domanda e offerta

Scheduling – 1

- Assegnamento di risorse alle attivita' sull'asse temporale
- Vari tipi di scheduling
  - Disjunctive scheduling
    - Ogni risorsa puo' eseguire solo 1 attivita' alla volta
  - Cumulative scheduling
    - Ogni risorsa puo' eseguire parecchie attivita' in parallelo purche' non sia superata la capacita' delle risorse
  - Non-preemptive scheduling
    - Le attivita' non possono essere interrotte
  - Preemptive scheduling
    - Le attivita' possono essere interrotte da altre attivita'

Scheduling – 2

- Job shop scheduling problem
  - Insieme di job
  - Insieme di macchine
  - Ogni job e' caratterizzato da un insieme di attivita' che devono essere processate in un dato ordine
  - Ogni attivita' e' definita da una durata e dalla macchina che la esegue
  - Una macchina puo' eseguire solo un'attivita' alla volta

Scheduling – 3

- Preemptive job shop scheduling problem
  - Job shop scheduling problem in cui tutte le attivita' sono interrompibili
  - Le attivita' possono essere interrotte
    - In ogni momento
    - Un numero illimitato di volte
  - Il problema
    - Trovare una schedulazione (cioe' un insieme di tempi di esecuzioni per ogni attivita') che minimizza il makespan (cioe' l'istante in cui tutte le attivita' sono finite)

Scheduling – 4

- Vedremo un algoritmo di Branch and Bound che si basa sulla propagazione con vincoli per risolvere il Disjunctive and Preemptive Job shop scheduling problem [Baptiste and Le Pape 1996]
  - Una macchina puo' eseguire solo un'attivita' alla volta
  - Le attivita' possono essere interrotte da altre attivita'

Il criterio di dominanza – 1

- Lo schema di branching utilizzato nell'algoritmo si basa sul criterio della dominanza che permette di ordinare le attivita' che utilizzano la stessa macchina
- **Definizione**
  - Per ogni schedulazione S e ogni attivita' A, definiamo la due date di A in S, $d_S(A)$ come
    - Il makespan di S, se A e' l'ultima attivita' del suo job
    - L'istante iniziale del successore di A, altrimenti
  - Diciamo che un'attivita' A_k ha priorita' sull'attivita' A_l in S ($A_k <_S A_l$) se e solo se
    - $d_S(A_k) <_S d_S(A_l)$ oppure
    - $d_S(A_k) = d_S(A_l)$ e $k <= l$

Il criterio di dominanza – 2

- **Teorema**
  - Per ogni schedulazione S, esiste una schedulazione J(S) tale che:
    - J(S) rispetta le due dates
      - $\forall$A, l'istante finale di A in J(S) e' al piu' $d_S(A)$
    - J(S) e' attiva
      - $\forall$M, $\forall$t, se qualche attivita' A$\in$ ACTS(M) e' disponibile al tempo t, M e' attiva al tempo t
      - 'Disponibile' significa che il predecessore di A e' finito e A non e' finito
    - J(S) segue il priority order $<_S$
      - $\forall$M, $\forall$t, $\forall$ $A_k \in$ ACTS(M), $A_k \neq A_l$, se A_k viene eseguita al tempo t
        - A_l non e' disponibile al tempo t, oppure
        - $A_k <_S A_l$

Jackson derivation – 1

- **Teorema**
  - Per ogni schedulazione S, esiste una schedulazione J(S) che:
    - rispetta le due dates
    - e' attiva
    - segue il priority order $<_S$
- J(S) viene costruito in modo cronologico
  - Per ogni istante t, su ogni macchina M, viene schedulata l'attivita' disponibile che e' piu' piccola nel priority order $<_S$
- J(S) viene chiamato 'Jackson derivation' di S

Jackson derivation – 2

makespan J(S) ≤ makespan S

almeno una schedulazione ottima e' una Jackson derivation di un'altra schedulazione

Nella ricerca della soluzione ottima imponiamo che abbia le caratteristiche di una Jackson derivation

Minimizzare il makespan

- Algoritmo di minimizzazione del makespan che sfrutta il risultato appena illustrato
  1. Calcola
     - un upper bound ovvio UB del makespan
     - un lower bound iniziale LB del makespan
  2. Seleziona un valore v in [LB,UB)
  3. Vincola il makespan ad essere minore o uguale v ed esegue la **strategia di branching**
     - Se viene trovata una soluzione, UB viene settato al makespan della soluzione
     - Se non viene trovata una soluzione (cioe' la ricerca fallisce), LB viene settato a v+1
  4. Itera i passi 2 e 3 finche' UB=LB

Strategia di branching – 1

- Strategia di branching che sfrutta il criterio di dominanza
  1. Sia t il primo istante in cui c'e' un'attivita' A disponibile (e non ancora schedulata).
  2. Calcola K = insieme delle attivita' disponibili al tempo t sulla stessa macchina di A
  3. Calcola NDK = insieme delle attivita' non 'dominate' in K
  4. Seleziona un'attivita' A_k in NDK (per es. quella con il piu' piccolo latest end time). Esegue A_k al tempo t e propaga la decisione e le sue conseguenze secondo il criterio di dominanza
     - Le altre attivita' di NDK vengono tenute come alternative da provare in seguito nella fase di backtrack
  5. Itera finche'
     1. tutte le attivita' sono schedulate oppure
     2. tutte le alternative sono state provate

Strategia di branching – 2

- Il potere di questa strategia di branching dipende molto dalle regole usate per
  - Eliminare le attivita dominate nel passo 3
  - Propagare le conseguenze della scelta di A_k nel passo 4
- Il criterio di dominanza viene sfruttato cosi'
  - Ogni volta che $A_k \in ACTS(M)$ viene eseguita al tempo t, o viene fatta eseguire
    - fino al suo earliest end time, oppure
    - fino all'earliest start time di un'altra attivita' $A_l \in ACTS(M)$ che non era disponibile al tempo t

Strategia di branching – 3

- Ogni volta che $A_k \in K$ viene eseguita al tempo t, ogni altra attivita' $A_l \in K$ puo' essere vincolata a non essere eseguita tra t e la fine di A_k
  - Al tempo t'>t questo riduce l'insieme dei candidati per l'esecuzione (A_l e' dominato da A_k)
  - Nel passo 4 si possono aggiungere dei vincoli ridondanti
    - $end(A_k) + rp_t(A_l) \leq end(A_l)$
      - $rp_t(A_l)$ e' il tempo di processamento rimanente di A_l
    - $end(A_k) \leq start(A_l)$, se A_l non e' iniziato al tempo t
- Se $A_k \in ACTS(M)$ e' l'ultima attivita' del suo job, A_k non e' candidata per l'esecuzione al tempo t, se in quel momento e' disponibile un'altra attivita' $A_l \in ACTS(M)$, che non e' l'ultima attivita' del suo job o tale che l<k (A_k e' dominata da A_l)

Unione europea
Fondo sociale europeo

MINISTERO DEL LAVORO
E DELLE POLITICHE SOCIALI
Direzione Generale per le Politiche
per l'Orientamento e la Formazione

REGIONE DEL VENETO

Schedulazione di attività in presenza di attività interrompibili

Maria Silvia Pini

Resp. accademico: Prof.ssa Francesca Rossi

Università di Padova

Attività FSE DGR 1102/2010

La gestione dell'informazione nell'azienda

Schedulazione intelligente di attività in presenza di risorse limitate e matching stabile ed efficiente tra domanda e offerta