CS341: Operating System

Scheduling Algorithms

Lect13 : 28th Aug 2014

Dr. A. Sahu
Dept of Comp. Sc. & Engg.
Indian Institute of Technology Guwahati

1

Outline

- Scheduling System Oriented
  - FCFS, SJF, Priority, RR
  - Multi Level Queue, MLQ with feedback
- Scheduling Algorithm
  - **Introduction to Scheduling Algorithms**
  - **Real Time Scheduling Algorithms**
  - **Multiprocessor Scheduling Algorithms**
  - **Distributed and Power Aware Scheduling**

2

Scheduling Problems

- In a scheduling problem
  - One has to find time slots in which activities should be processed under given constraints.
- The main constraints are
  - Resource constraints and
  - Precedence constraints between activities
- A quite general scheduling problem is
  - ***Resource Constrained Project Scheduling Problem***
  - In short **RCPSP**

Parallel Machine Problems

- For **identical machines $M_1, \ldots, M_m$**
  - The processing time for ***j*** is the same on each machine.
- For **unrelated machines**
  - The processing time p_{jk} depends on the machine M_k on which j is processed.

Parallel Machine Problems

- For **uniform machine**
  - if $p_{jk} = p_j/r_k$.
- For **multi-purpose machines**
  - A set of machines μ_j is is associated with each job j indicating that j can be processed on one machine in μ_j only.

Example: Machine Environment

	M1	M2	M3
P1	5	5	5
P2	8	8	8
P3	6	6	6

Identical

	M1	M2	M3
P1	5	4	6
P2	9	8	4
P3	3	18	4

Unrelated

	M1	M2	M3
P1	5	5/1.5	5/2
P2	9	9/1.5	9/2
P3	9	9/1.5	9/2

Uniform

6

Classification of Scheduling Problems

Classes of scheduling problems can be specified in terms of the three-field classification

$$\alpha \mid \beta \mid \gamma$$

where

- α specifies the **machine environment**,
- β specifies the **job characteristics**, and
- γ describes the **objective function(s).**

Machine Environment

- 1 single machine
- P parallel identical machines
- Q uniform machines
- R unrelated machines
- MPM multipurpose machines, J job-shop,
- F flow-shop O open-shop

If the number of machines is fixed to m we write **Pm, Qm, Rm,** MPMm, Jm, Fm, Om.

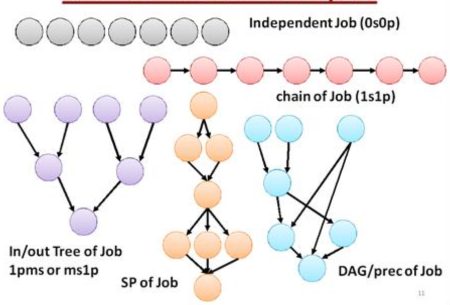
Job Characteristics

- **pmtn** preemption
- r_j release times /arrival time
- d_j deadlines
- $p_j = 1$ **or** $p_j = p$ **or** $p_j \in \{1,2\}$ restricted processing times

Job Characteristics

- **prec** arbitrary precedence constraints
- **intree (outtree)** intree (or outtree) precedences
- **chains** chain precedences
- **series-parallel** a series-parallel precedence graph

Job Precedence Examples



Objective Functions

Two types of objective functions are most common:

- **bottleneck objective functions** $\max\{f_j(C_j) \mid j = 1, \dots, n\}$, and
- **sum objective functions** $\Sigma f_j(C_j) = f_1(C_1) + f_2(C_2) + \dots \dots + f_n(C_n)$.

Objective Functions

- C_{max} and L_{max} symbolize the bottleneck objective functions with
  - $f_j(C_j) = C_j$ (makespan)
  - $f_j(C_j) = C_j - d_j$ (maximum lateness)
- Common sum objective functions are:
  - ΣC_j (mean flow-time)
  - $\Sigma \omega_j C_j$ (weighted flow-time)

Objective Functions

- **Number of Late Job**
  - ΣU_j (number of late jobs) and $\Sigma \omega_j U_j$ (weighted number of late jobs) where $U_j = 1$ if $C_j > d_j$ and $U_j = 0$ otherwise.
- **Tardiness**
  - ΣT_j (sum of tardiness) and $\Sigma \omega_j T_j$ (weighted sum of tardiness)
  - Tardiness of job **j** is given by

$$T_j = \max \{ 0, C_j - d_j \}.$$

Examples

- $1 \mid prec; p_j = 1 \mid \Sigma \omega_j C_j$
- $P2 \mid \mid C_{max}$
- $P \mid p_j = 1; r_j \mid \Sigma \omega_j U_j$
- $R2 \mid chains; pmtn \mid C_{max}$
- $P3 \mid n = 3 \mid C_{max}$
- $Pm \mid p_{ij} = 1; outtree; r_j \mid \Sigma C_j$

Example: $1|C_{max}$

- N independent job without pre-emption
- 1 processor
- Minimize Cmax
- Sol: Schedule in any orders

16

Example: $1|\sum C_i$

- N independent job without pre-emption
- 1 processor
- Minimize $\sum C_i$
- Sol: Schedule shortest processing time first
  - SJF is optimal

17

Example: $1|\sum w_i C_i$

- N independent job without pre-emption
- 1 processor
- Minimize $\sum w_i C_i$
- Sol:
  - Calculate processing time to weight ratio
  - Rank jobs in increasing order of p_i/w_i and schedule accordingly
  - The Weighted Shortest Processing Time First rule is Optimal for $\mathbf{1|\sum w_i C_i}$

18

Example: 1|chain|$\sum w_iC_i$

- N independent jobs with chain precedence without pre-emption
- 1 processor, multiple chain
- Minimize $\sum w_iC_i$
- Sol:
  - Calculate processing time to weight ratio (ρ) of chains (by including a number of tasks from a chains)
  - Process the tasks from chain till the ρ of the chain is higher than others chain

19

Example: 1|prec|$\sum w_iC_i$

- For general precedence the problem is Hard
- NP-Complete problem

20

Example: 1||$\sum T_i$

- All job are independent job
- Each job associated with two things
  - Execution time p_i and deadline D_i
- Tardiness is T_i=max{0, C_i-D_i}, where C_i is completion time of Task *i*
- Optimality & Optimal Structure
  - If $p_j \le p_k$ && $d_j \le d_k$ then there exist an optimal sequence in which job *j* is scheduled before job *k*
- Dynamic Programming: Left as exercise

21

P3|ptmn|C_{max}

- 3 Identical machine, Independent Jobs, release time ri=0, C_{max}
- Solvable in Polynomial time
- Suppose N tasks with execution time t_i, 3 processor
- C_{max}= ($\sum t_i$)/3
- Distribute C_{max} unit amount task to each processor in any order

22

P3|ptmn|C_{max}

- M1, M2, M3
- 10 tasks: 5,6,10,4,3,8,6,3,7,12
- C_{max}= (5+6+10+4+3+8+6+3+7+12)/3=64/3=21+1/3
- Assign 20+1/3 unit time of tasks in any ways

  12+6+3 +1/3

  10+7+4 +1/3

  8+5+6+ 2 +1/3

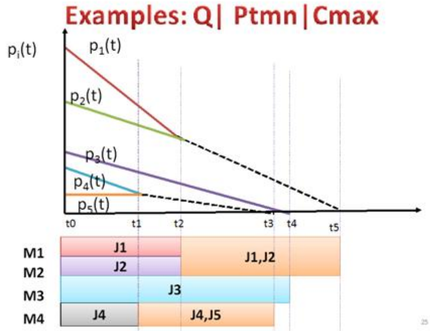
23

Q|ptmn|C_{max}

- Suppose 5 Job p1, p2, p3, p4 and p5
  - With P1 is longest and P5 is shortest
  - P1>p2>p3>p4>p5
- 4 machine M1 > M2 > M3 > M4. M1 is fastest and M4 is slowest
- Each job have level : based on how time left before finish (un-finished part of job).
- Fist try to finish the highest level job on the fastest machine.
- When level of two job are same jointly process on the machine

24

Examples: Q| Ptmn|Cmax



Q|ptmn|C_{max}

Algorithm level

1. *t := 0;*
2. WHILE there exist jobs with positive level DO {
3. Assign(*t);*
4. *t1 := min{s > t | a job completes at time s};*
5. *t2 := min{s > t | there are jobs i, j with pi(t) > pj(t) and pi(s) = pj(s)};*
6. *t := min{t1, t2}*
7. Construct the schedule.

Q|ptmn|C_{max}

Assign (t) {

1. *J := {i | pi(t) > 0};*
2. *M := {M1, . . .,Mm};*
3. WHILE *J = Ø and M = Ø DO {*
4. Find the set *I ⊆ J of jobs with highest level;*
5. *r := min{|M|, |I|};*
6. Assign jobs in *I to be processed jointly on the r fastest* machines in *M;*
7. *J := J\I;*
8. Eliminate the *r fastest machines in M from M*

Discrete time: Q|ptmn|C_{max}

- Suppose preemption is allowed only at boundary or some discrete point…
- Longest Remaining Time on Fasted Machine (LRT-FM) yield optimal schedule for **Q|ptmn|C_{max}** in discrete time
- *Proof: left…. Scheduling Book By M. Pindo*

P_m||C_{max}

- 2 || Cmax can be easily mapped to 2 set partition problem
  - Pseudo Polynomial Time algorithm
- m≥3, m=3 mapped to 3 partition problem
  - NP Complete Problem
- Approximation Algorithms
  - Grahams List scheduling
  - Longest Task First

Approximation for: Pm|prec,pi=1|C_{max} and Pm||C_{max}

- **CP Algorithms: Introduction to Algorithms, Corman Leisserson Rivest (CLR) , 3rd Ed, Page 779-783**
- ***Algorithm Design, Eva Tardos, Page 600-605,***