Lecture 2: Scheduling on Parallel Machines

Loris Marchal

(reminders on scheduling on 1-machine, questions)

1 Scheduling on Parallel Machines

- P parallel identical
- Q uniform machines: each machine has a given speed $speed_i$, and all jobs have a size $size_j$, the processing time is given by $size_j/speed_i$
- R unrelated machines: the processing time of job j on machine i is given by $p_{i,j}$, without any other constraints

Sometimes the number of processors is fixed: for example, P2, F2, or Pm.

1.1 P$||C_{\max}$

- NP-completeness: straightforward reduction to Partition (weakly NP-complete) or 3-Partition (unary NP-complete)
- List-scheduling (Graham): $2 - 1/m$-approx
- LPT $(4/3 - 1/3m)$-approx, tight

Details in http://www.seas.upenn.edu/~deepc/Courses/CO454/Lectures/lecture12.pdf

1.2 P|prec|$C_{\max}$

- precedence constraints: $i \to j$ means that j cannot start before i completes
- often modeled using a Directed Acyclic Graph
- any path of precedence is a lower bound on the optimal makespan
- critical path: (one of) the longest path
- precedence may have special structure:
  - prec : arbitrary precedence constraints
  - intree: (outtree) intree (or outtree) precedences

– chains: chain precedences

Adaptation of the Graham list-scheduling approximation ratio.

Details in http://www.seas.upenn.edu/~deepc/Courses/CO454/Lectures/lecture13.pdf

2 Shop scheduling problems

- jobs: $J_1, \ldots, J_m$
- processors: $P_1, \ldots, P_m$
- job J_j consists in operations $J_{1,j}, J_{2,j}, \ldots, J_{n_j,j}$
- operation $J_{i,j}$ takes time $p_{i,j}$ and must be processed by machine $\mu_{i,j}$

	no precedence	precedence chains
arbitrary structure		J: job-shop
$\forall j, n_j = m, \mu_{i,j} = P_i$	O: open-shop	F: flow-shop

2.1 List-scheduling for O$||C_{\max}$

We prove that any list-scheduling algorithm is a 2-approximation for O$||C_{\max}$ first, two simple bounds on the optimal makespan:

- $\forall j, \quad OPT \geq \sum_{i=1}^{m} p_{i,j}$ (all operations of each job must be processed)
- $\forall i, \quad OPT \geq \sum_{j=1}^{n} p_{i,j}$ (each machine must process all required operations)

Let (M, l) be the pair machine/job which finishes the last one. At any time step before $C_{\max}$, either M is busy processing something or l is busy being processed by some machine, so:

$$C_{\max} \leq \sum_{j=1}^{n} P_{M,j} + \sum_{i=1}^{m} P_{i,l} \leq 2OPT$$