Flow Shop for Dual CPUs in Dynamic Voltage Scaling

Vincent Chau, Ken C.K. Fong, **Minming Li** and Kai Wang

Department of Computer Science, City University of Hong Kong

March 2016

香港城市大學
City University
of Hong Kong

Outline

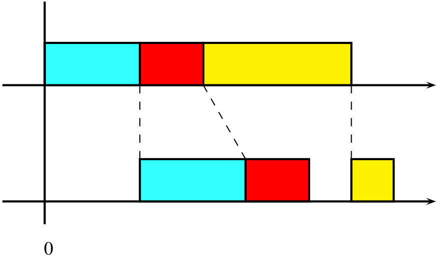
City University of Hong Kong
香港城市大學

Outline

香港城市大學
City University of Hong Kong

Models

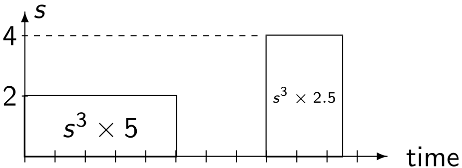
- We are given a set of n jobs and m machines:
  - each job j has a processing requirement $p_{i,j}$ on machine i
- Flowshop on 2 machines
  - a job j can start on machine 2 only when it is completed on machine 1



Models

- We are given a set of n jobs and m machines:
  - each job j has a processing requirement $p_{i,j}$ on machine i
- Flowshop on 2 machines
  - a job j can start on machine 2 only when it is completed on machine 1
- Speed-Scaling setting
  - Cost is $\int s(t)^{\alpha} dt$ with $\alpha > 1$

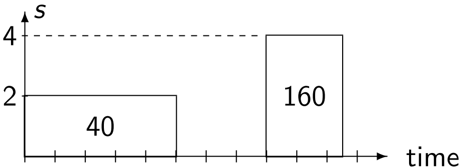
$p = 10$
$\alpha = 3$



Models

- We are given a set of n jobs and m machines:
  - each job j has a processing requirement $p_{i,j}$ on machine i
- Flowshop on 2 machines
  - a job j can start on machine 2 only when it is completed on machine 1
- Speed-Scaling setting
  - Cost is $\int s(t)^\alpha dt$ with $\alpha > 1$

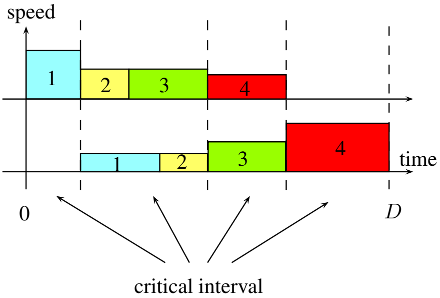
$p = 10$
$\alpha = 3$



Example

When order of jobs is given, there exists a $O(n^3)$ algorithm
Z. Mu, M. Li, Journal of Combinatorial Optimization, 2015

$$E = \frac{\left(p_{1,1} + \sqrt[\alpha]{(p_{1,2} + p_{1,3})^\alpha + (p_{2,1} + p_{2,2})^\alpha} + \sqrt[\alpha]{p_{1,4}{}^\alpha + p_{2,3}{}^\alpha} + p_{2,4}\right)^\alpha}{D}$$



Our contributions

- Flowshop on m machines
  - Fixed order, Discrete speeds, a Linear Program Formulation
  - Arbitrary order, Continuous speeds, an approximation algorithm
- Sense-And-Aggregate Model

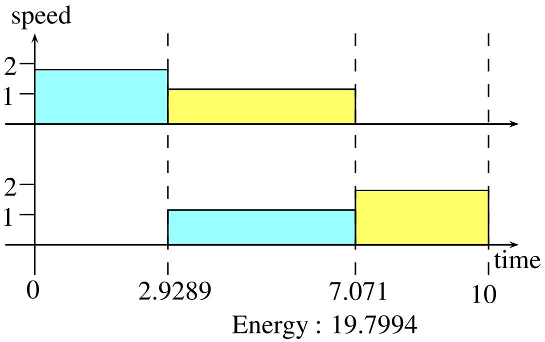
香港城市大學
City University of Hong Kong

Outline

1. Introduction
2. Flowshop on m machines
   - Discrete Speed (fixed order)
   - Continuous Speed (arbitrary order)
3. Sense-And-Aggregate Model
4. Conclusion

Continuous to Discrete

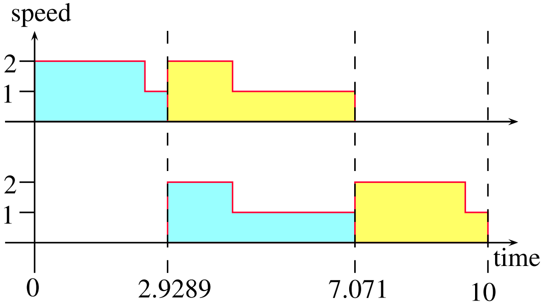
- Jobs order is given with processing requirement of 10
- Set of speeds $S = \{1, 2\}$



$\alpha = 2$

Continuous to Discrete

- Jobs order is given with processing requirement of 10
- Set of speeds $S = \{1, 2\}$



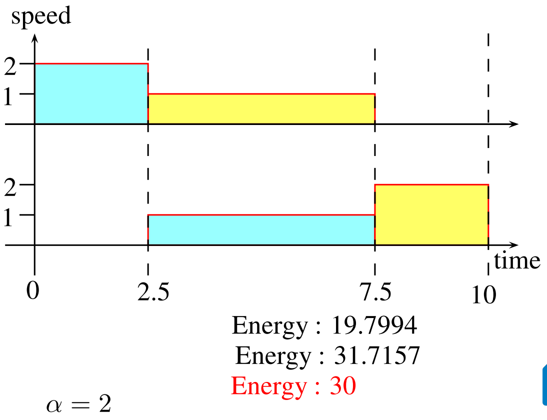
Energy : 19.7994

Energy : 31.7157

$\alpha = 2$

Continuous to Discrete

- Jobs order is given with processing requirement of 10
- Set of speeds $S = \{1, 2\}$



A Linear Program

Let $x_{i,j,v}$ be the workload done for job j on machine i at speed v. Let $s_{i,j}$ (resp. $c_{i,j}$) be the starting time (resp. completion time) of job j on machine i.

$$\min \quad \sum_{v \in S} \sum_{i} \sum_{j} v^{\alpha-1} x_{i,j,v}$$

$$\begin{aligned}
\text{s.t.} \sum_{v \in S} x_{i,j,v} &= p_{i,j} && \forall i,j && \text{all jobs must be scheduled} \\
s_{i,j} + \sum_{v \in S} \frac{x_{i,j,v}}{v} &= c_{i,j} && \forall i,j && \text{Processing time} \\
c_{m,n} &\leq D && && \\
c_{i,j} &\leq s_{i,j+1} && \forall i,j && \text{Precedence const. between jobs} \\
c_{i,j} &\leq s_{i+1,j} && \forall i,j && \text{between machines} \\
x_{i,j,v},\ s_{i,j},\ c_{i,j} &\geq 0 && \forall i,j,v &&
\end{aligned}$$

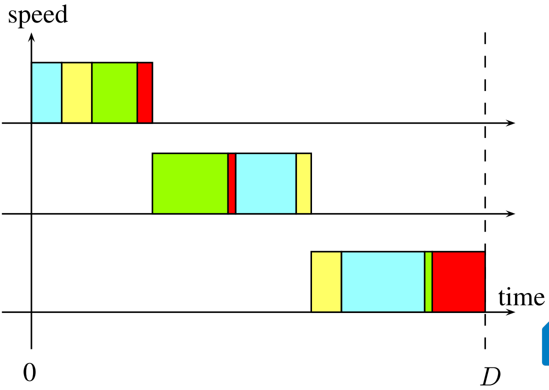
A Linear Time Approximation Algorithm

- Recall that for fixed order, a polynomial time algorithm exists

香港城市大學
City University of Hong Kong

A Linear Time Approximation Algorithm

- Recall that for fixed order, a polynomial time algorithm exists
- We schedule jobs at speed $\frac{\sum_{i,j} p_{i,j}}{D}$ in any order on each machine



Approximation algorithm

Theorem

This algorithm is a $m^{\alpha-1}$-approximation

Approximation algorithm

Theorem

This algorithm is a $m^{\alpha-1}$-approximation

Proof

Let $V_i = \sum_j p_{i,j} \; \forall i$

$$\frac{ALG}{LB} = \frac{(\sum_i V_i)^\alpha D^{1-\alpha}}{m\left(\sum_i \frac{V_i}{m}\right)^\alpha D^{1-\alpha}} = \frac{(\sum_i V_i)^\alpha}{m\left(\sum_i \frac{V_i}{m}\right)^\alpha}$$

$$= \frac{(\sum_i V_i)^\alpha}{m(\sum_i V_i)^\alpha \left(\frac{1}{m}\right)^\alpha} = m^{\alpha-1}$$

香港城市大學
City University
of Hong Kong

Approximation algorithm

Theorem

This algorithm is a $m^{\alpha-1}$-approximation

Proof

Let $V_i = \sum_j p_{i,j} \ \forall i$

$$\frac{ALG}{LB} = \frac{(\sum_i V_i)^\alpha D^{1-\alpha}}{m\left(\sum_i \frac{V_i}{m}\right)^\alpha D^{1-\alpha}} = \frac{(\sum_i V_i)^\alpha}{m\left(\sum_i \frac{V_i}{m}\right)^\alpha} = \frac{(\sum_i V_i)^\alpha}{m(\sum_i V_i)^\alpha \left(\frac{1}{m}\right)^\alpha} = m^{\alpha-1}$$

Note that if we fix an arbitrary order and compute the minimum energy consumption, the approximation cannot be larger than $m^{\alpha-1}$ but takes $O(n^3)$ time.

香港城市大學
City University of Hong Kong

Outline

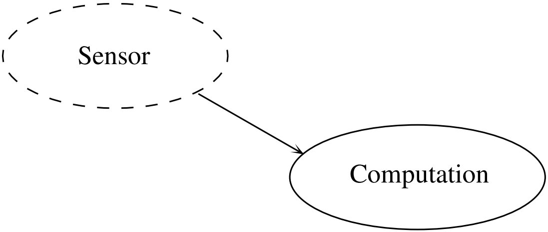
1. Introduction
2. Flowshop on m machines
   - Discrete Speed (fixed order)
   - Continuous Speed (arbitrary order)
3. Sense-And-Aggregate Model
4. Conclusion

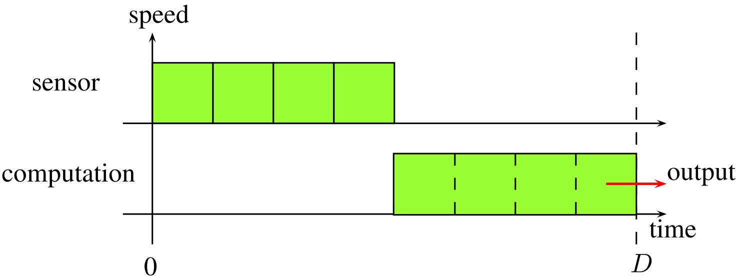
香港城市大學 City University of Hong Kong

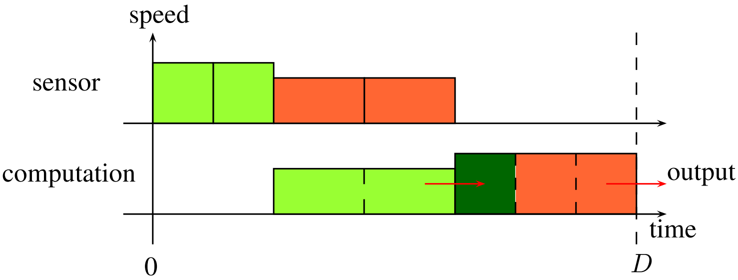
Sense-And-Aggregate Model

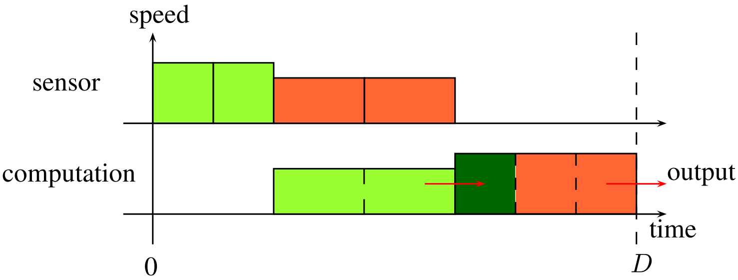
Rules:

- Sensor collects one unit of data at each time
- Computation can decide to process now or wait for more data
  - Outputs one unit of data for each aggregation
- Common deadline D




speed
sensor
computation
output
time
0
D


speed
sensor
computation
output
time
0
D



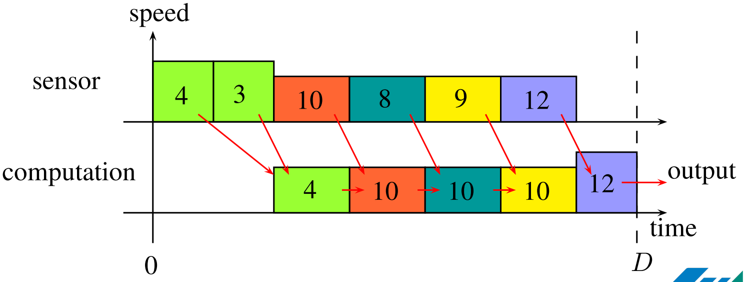
Observations

- The more we wait, the less workload there is on the computation machine
- Decide to compute earlier allows to speed down the processing, and potentially the energy consumption

香港城市大學 City University of Hong Kong

Workload-Consideration-Function

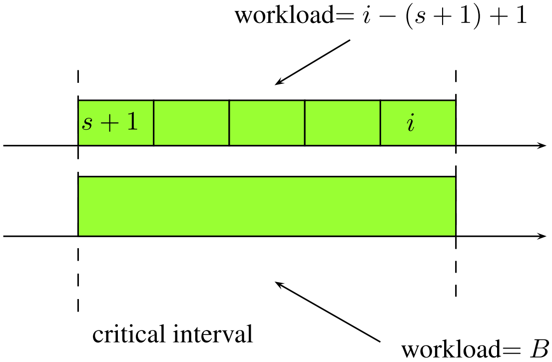
- The computation depends on the nature of the problem
- For example: we want the maximum/minimum value: $f(x) = x - 1$



Optimal schedule: Compute as soon as possible

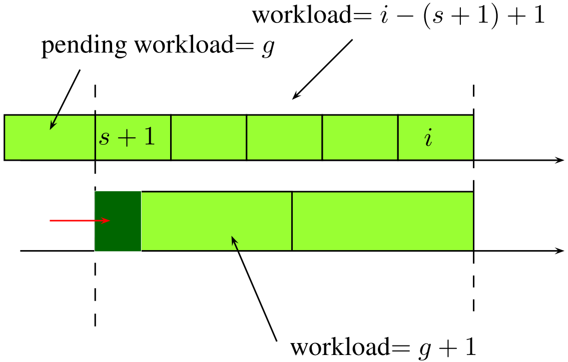
Workload-Consideration-Function $f(x) = x$

- Guess the critical intervals: the workload of each machine
- Guess when to start each computation/aggregation



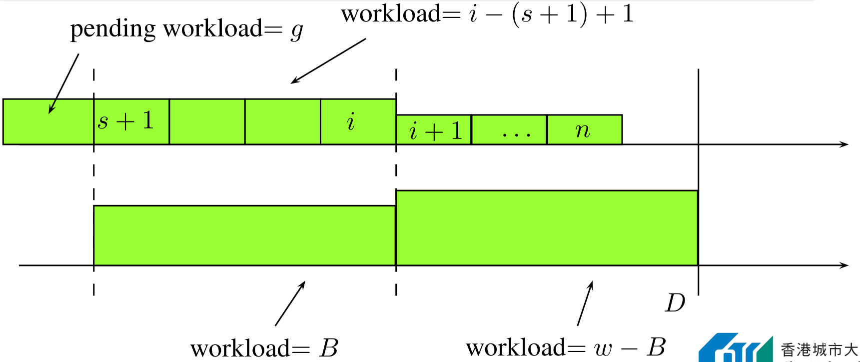
Workload-Consideration-Function $f(x) = x$

- Guess the critical intervals: the workload of each machine
- Guess when to start each computation/aggregation



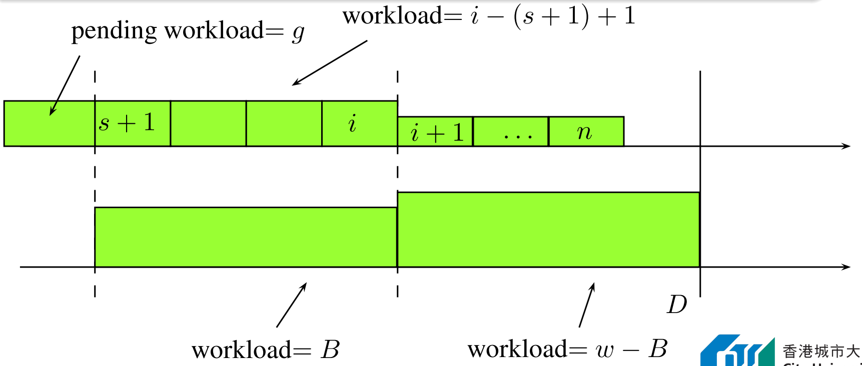
Definition

Let $F(s, w, g)$ be the minimum cost of the jobs $s+1, \ldots, n$ with a workload of w on the second machine and a pending workload of g on the first machine (before job $s+1$).



Definition

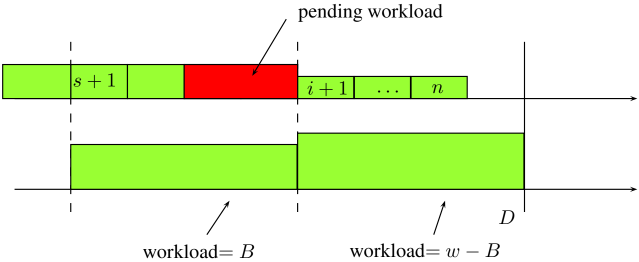
Let $F(s, w, g)$ be the minimum cost of the jobs $s+1, \ldots, n$ with a workload of w on the second machine and a pending workload of g on the first machine (before job $s+1$).



We need to guess the value of B and i at each step.

Dynamic Programming

Once the values i and B are fixed, we need to compute the minimum pending workload in this critical interval.

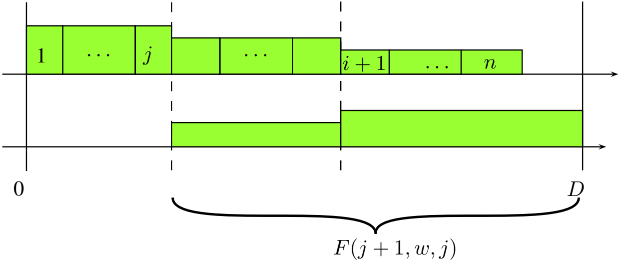


$$F(s,w,g) = \min_{\substack{s+1\le i\le n \\ 1\le B\le w}} \sqrt[\alpha]{(i-s)^\alpha + B^\alpha} + F(i+1, w-B, (i-s)-k)$$

where k is the pending workload

香港城市大學
City University
of Hong Kong

Objective function



The cost of the optimal schedule is

$$\min_{\substack{0 \le w \le W \\ 1 \le j \le n}} \frac{(F(j+1, w, j) + j)^{\alpha}}{D}$$

Note that when workloads are fixed during the computation, the time of critical intervals are also fixed.

Guessing the minimum pending workload

Definition

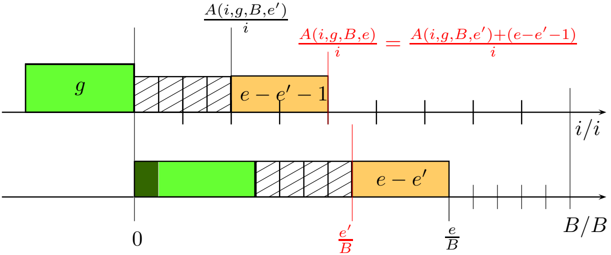
$A(i, g, B, e)$ is the maximum workload on the first machine that can be aggregated such that:

- there is at most a workload of i on the first machine
- there is at most a workload of B on the second machine
- there is a pending workload of g (already scheduled on the first machine on a previous critical interval)
- the second machine has already scheduled a workload of e

The remaining workload is the pending workload:
$k = (i - s) - A(i, g, B, B)$

Dynamic Programming (2)

In this critical interval, we ensure precedence constraints.
$\frac{A(i,g,B,e')+(e-e'-1)}{i}$ should be before $\frac{e'}{B}$ (from the beginning of the interval)



$$A(i,g,B,e) = \max \begin{cases} A(i,g,B,e-1) \text{ (cannot aggregate)} \\ \max\limits_{\substack{0 \le e' < e \\ \frac{A(i,g,B,e')+(e-e'-1)}{i} \le \frac{e'}{B}}} A(i,g,B,e') + (e-e'-1) \end{cases}$$

$$A(i,g,B,0) = -g$$

香港城市大學
City University of Hong Kong

To sum up

- When $f(x) = x$, then $B \leq w \leq 2n$ which lead to an overall time complexity of $O(n^5)$.
- When $f(x) = x - 1$, a greedy algorithm in linear time can solve it.
- Other workload-consideration-function $f(x)$?
  - Can solve any function $f(x)$ in time $O(n^3 W^2)$
  - where $W \leq n(\max_{0 \leq x \leq n} f(x) + 1)$
  - Overall complexity : $O(n^5 (\max f(x))^2)$

香港城市大學 City University of Hong Kong

Outline

香港城市大學
City University
of Hong Kong

Conclusion and Directions

- Flowshop on m machines
  - Fixed order, Discrete speeds, a Linear Program Formulation
    - A more efficient algorithm?
  - Arbitrary order, Continuous speeds, an approximation algorithm
    - Improve the approximation ratio
    - open : Is the 2-machine-flowshop polynomial when order is not fixed?
- Sense-And-Aggregate Model
  - A more general workload-consideration-function
  - Approximation algorithm
  - Online setting

香港城市大學 City University of Hong Kong

Thanks for your attention!

香港城市大學
City University
of Hong Kong