Divisible load theory

Loris Marchal

November 13, 2012.

1 Summary of the first lecture

- Key assumption : work is *divisible* in rational quantities
- Renders many problems tractable
- Underlying idea : since the number of tasks is large, rounding to integer values after computing optimal schedule is negligible

- A schedule is described by :
  - the set of participating processor,
  - the order of the sending operations,
  - the quantity of work sent to each processor

- Basic problem : star network, linear costs, one-round,
  - All workers participate
  - Send work to workers with largest bandwidth first
  - All workers terminate at the same time : we are able to compute the amount of work done by each worker

Many possible generalizations.

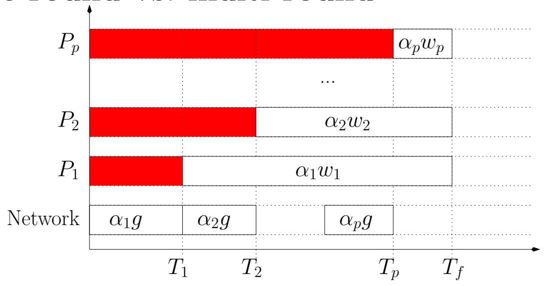
2 Adaptation to tree-shaped platforms

- Each single level tree can be replaced by a single node, with total computing capacity W, with $w = \sum \alpha_i$, where α is the solution of the previous linear program
- Constructive solution for the tree :
  1. Traverse the tree from bottom to top, replacing each single-level node by a equivalent processor
  2. Solve the star problem obtained
  3. Traverse the tree from top to bottom, undo each transformation, order the children, and distribute the load.
- Global solution : order the children by non-decreasing bandwidth, and then write the complete linear program.

3 Multi-round algorithms

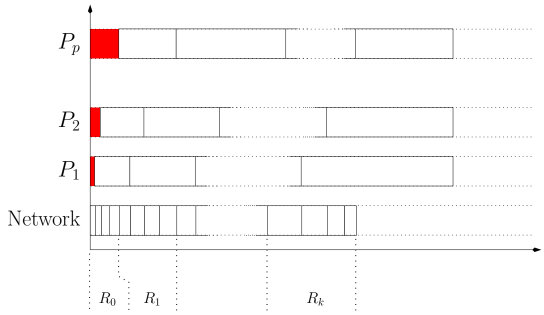
3.1 Discussion

One round vs. multi-round



One round

$\rightsquigarrow$ long idle-times



Multi-round

Efficient when W_{total} large

Intuition : Start with small rounds, then increase chunk sizes.

Problem with current model : leads to an absurd solution with infinite number of infinitely small messages.

- Either change the model in order to allow simultaneous communication and computation on the same data
- Or add latency to the communication the model

Notations

- A set P_1, ..., P_p of processors
- P_1 is the master processor : initially, it holds all the data.
- The overall amount of work : W_{total}.
- Processor P_i receives an amount of work $\alpha_i W_{\text{total}}$ with $\sum_i n_i = W_{\text{total}}$ with $\alpha_i W_{\text{total}} \in \mathbb{Q}$ and $\sum_i \alpha_i = 1$.
  Length of a unit-size work on processor P_i : w_i.
  Computation time on P_i : $n_i w_i$.
- **Time needed to send a message of size α_i P_1 to P_i : $L_i + c_i \times \alpha_i$.**
  One-port model : P_1 sends and receives a *single* message at a time.

Complexity

One round, $\forall i, c_i = 0$ Given W_{total}, p workers, $(P_i)_{1 \le i \le p}$, $(L_i)_{1 \le i \le p}$, and a rational number $T \geq 0$, and assuming that bandwidths are infinite, is it possible to compute all W_{total} load units within T time units ?

Theorem: The problem with one-round and infinite bandwidths is NP-complete.

What is the complexity of the general problem with finite bandwidths and several rounds ?

The general problem is NP-hard, but does not appear to be in NP (no polynomial bound on the number of activations).

3.2 Resolution with fixed sequence

Fixed activation sequence

Hypotheses

1. Number of activations : N_{act} ;
2. Whether P_i is **the** processor used during activation j : $\chi_i^{(j)}$

$$\textsc{Minimize } T, \textsc{ under the constraints}$$
$$\begin{cases} \displaystyle\sum_{j=1}^{N_{\text{act}}} \sum_{i=1}^{p} \chi_i^{(j)} \alpha_i^{(j)} = W_{\text{total}} \\ \forall k \leq N_{\text{act}}, \forall l : \left(\displaystyle\sum_{j=1}^{k} \sum_{i=1}^{p} \chi_i^{(j)} (L_i + \alpha_i^{(j)} c_i) \right) + \displaystyle\sum_{j=k}^{N_{\text{act}}} \chi_l^{(j)} \alpha_l^{(j)} w_l \leq T \\ \forall i, j : \alpha_i^{(j)} \geq 0 \end{cases} \tag{1}$$

Can be solved in polynomial time.

Fixed number of activations

$$\textsc{Minimize } T, \textsc{ under the constraints}$$
$$\begin{cases} \displaystyle\sum_{j=1}^{N_{\text{act}}} \sum_{i=1}^{p} \chi_i^{(j)} \alpha_i^{(j)} = W_{\text{total}} \\ \forall k \leq N_{\text{act}}, \forall l : \left(\displaystyle\sum_{j=1}^{k} \sum_{i=1}^{p} \chi_i^{(j)} (L_i + \alpha_i^{(j)} c_i) \right) + \displaystyle\sum_{j=k}^{N_{\text{act}}} \chi_l^{(j)} \alpha_l^{(j)} w_l \leq T \\ \forall k \leq N_{\text{act}} : \displaystyle\sum_{i=1}^{p} \chi_i^{(k)} \leq 1 \\ \forall i, j : \chi_i^{(j)} \in \{0, 1\} \\ \forall i, j : \alpha_i^{(j)} \geq 0 \end{cases} \tag{2}$$

Exact but exponential

Can lead to branch-and-bound algorithms

3.3 Uniform multi-round (UMR)

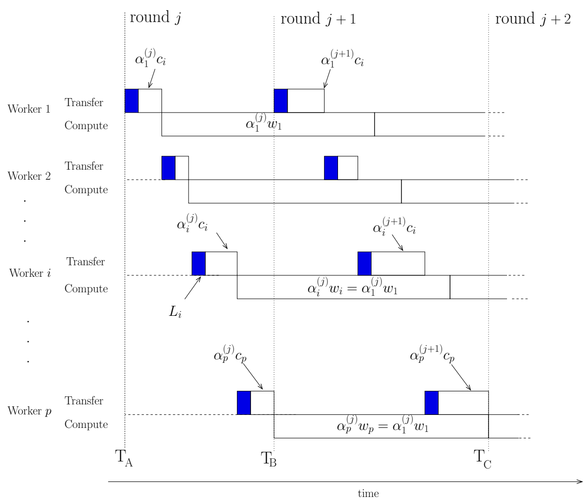
In a round : all workers have same computation time

Geometrical increase of rounds size

No idle time in communications :

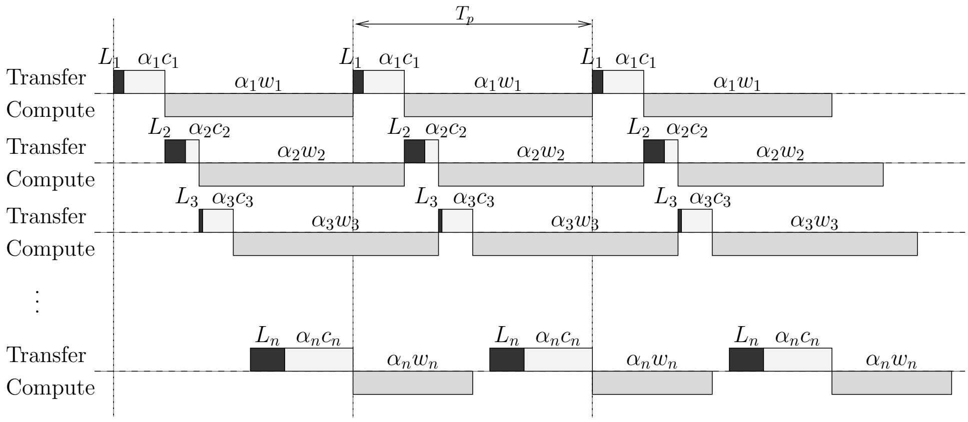
$$\alpha_i^{(j)} w_i = \sum_{k=1}^{p} (L_k + \alpha_k^{(j+1)} c_k).$$

Heuristic processor selection : by decreasing bandwidths



No guarantee...

3.4 Periodic schedule



How to choose T_p ? Which resources to select ?

Without overlap

Equations

- Divide total execution time T into k periods of duration T_p.
- $\mathcal{I} \subset \{1, \ldots, p\}$ participating processors.
- Bandwidth limitation :

$$\sum_{i \in \mathcal{I}} (L_i + \alpha_i c_i) \leq T_p.$$

- No overlap :

$$\forall i \in \mathcal{I}, \quad L_i + \alpha_i (c_i + w_i) \leq T_p.$$

Normalization

- β_i average number of tasks processed by P_i during one time unit.

– Linear program :

$$\text{MAXIMIZE} \sum_{i=1}^{p} \beta_i$$
$$\begin{cases} \forall i \in \mathcal{I}, \quad \beta_i(c_i + w_i) \leq 1 - \frac{L_i}{T_p} \\ \sum_{i \in \mathcal{I}} \beta_i c_i \leq 1 - \frac{\sum_{i \in \mathcal{I}} L_i}{T_p} \end{cases}.$$

– Easier version (more constrained) :
  – change second member of first constraints
  – sum in last constraint concerns all processors

$$\text{MAXIMIZE} \sum_{i=1}^{p} x_i$$
$$\begin{cases} \forall 1 \leq i \leq p, \quad x_i(c_i + w_i) \leq 1 - \frac{\sum_{i=1}^{p} L_i}{T_p} \\ \sum_{i=1}^{p} x_i c_i \leq 1 - \frac{\sum_{i=1}^{p} L_i}{T_p} \end{cases}.$$

Bandwidth-centric solution

– Sort : $c_1 \leq c_2 \leq \ldots \leq c_p$.
– Let q be the largest index so that $\sum_{i=1}^{q} \frac{c_i}{c_i+w_i} \leq 1$.
– If $q < p$, $\epsilon = 1 - \sum_{i=1}^{q} \frac{c_i}{c_i+w_i}$.
– Optimal solution to relaxed program :

$$\forall 1 \leq i \leq q, \quad x_i = \frac{1 - \frac{\sum_{i=1}^{p} L_i}{T_p}}{c_i + w_i}$$

and (if $q < p$) :

$$x_{q+1} = \left(1 - \frac{\sum_{i=1}^{p} L_i}{T_p}\right)\left(\frac{\epsilon}{c_{q+1}}\right),$$

and $x_{q+2} = x_{q+3} = \ldots = x_p = 0$.

Asymptotic optimality

– Let $T_p = \sqrt{T^*_{\max}}$ and $\alpha_i = x_i T_p$ for all i.
– Then $T \leq T^*_{\max} + O(\sqrt{T^*_{\max}})$.
– Closed-form expressions for resource selection and task assignment provided by the algorithm.

With overlap

Key points
– Still sort resources according to the c_i.
– Greedily select resources until the sum of the ratios $\frac{c_i}{w_i}$ $\left(\text{instead of } \frac{c_i}{c_i+w_i}\right)$ exceeds 1.

4 With bounded memory

Divisible load scheduling with bounded memory

– Assume the memory is bounded on each worker
– Problem is NP-complete with affine costs (reduction from 3-partition)