Divisible load theory

Frédéric Vivien

e-mail: Frederic.Vivien@inria.fr

September 18, 2014

Overview

Overview

Context of the study

- Scientific computing: large needs in computation or storage resources.
- Need to use systems with "several processors":
  - Parallel computers with shared memory.
  - Parallel computers with distributed memory.
  - Clusters.
  - Heterogeneous clusters.
  - Clusters of clusters.
  - Network of workstations.
  - The Grid.
- Problematic: to take into account the heterogeneity at the algorithmic level.

New platforms, new problems

Execution platforms: Distributed heterogeneous platforms (network of workstations, clusters, clusters of clusters, grids, etc.)

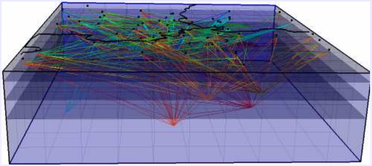
New sources of problems

- Heterogeneity of processors (computational power, memory, etc.)
- Heterogeneity of communications links.
- Irregularity of interconnection network.
- Non dedicated platforms.

We need to adapt our algorithmic approaches and our scheduling strategies: new objectives, new models, etc.

An example of application: seismic tomography of the Earth

- Model of the inner structure of the Earth



- The model is validated by comparing the propagation time of a seismic wave in the model to the actual propagation time.
- Set of all seismic events of the year 1999: $817,101$
- Original program written for a parallel computer:

```
if (rank = ROOT)
   raydata ← read n lines from data file;
MPI_Scatter(raydata, n/P, ..., rbuff, ...,
            ROOT, MPI_COMM_WORLD);
compute_work(rbuff);
```

Applications covered by the divisible load model

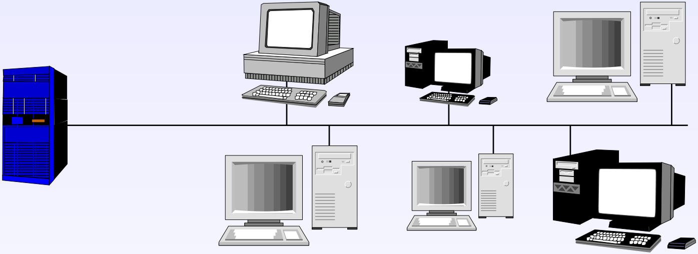
Applications made of a very (very) large number of fine grain computations.

Computation time proportional to the size of the data to be processed.

Independent computations: neither synchronizations nor communications.

Overview

Bus-like network



- The links between the master and the slaves all have the same characteristics.
- The slave have different computation power.

Notations

- A set P_1, ..., P_p of processors

Notations

- A set P_1, ..., P_p of processors
- P_1 is the master processor: initially, it holds all the data.

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}.

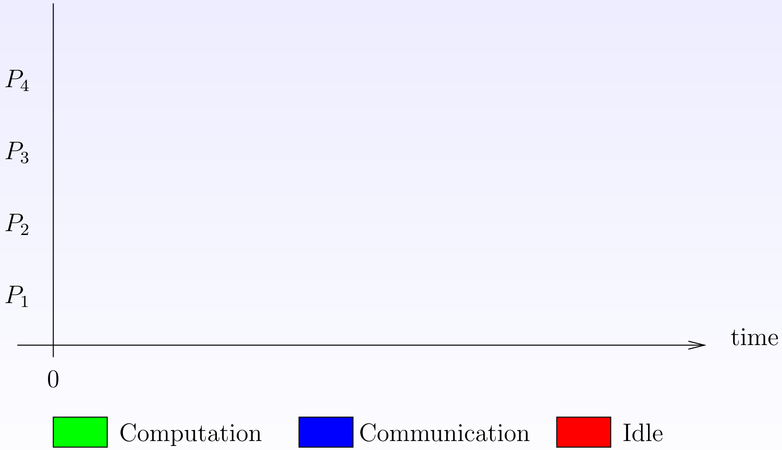
Notations

- 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: $n_i \in \mathbb{N}$ with $\sum_i n_i = W_{\text{total}}$.
  Length of a unit-size work on processor P_i: w_i.
  Computation time on P_i: $n_i w_i$.

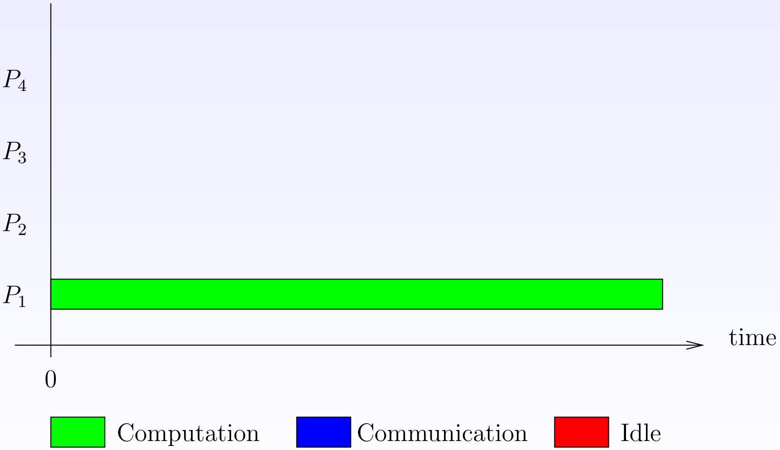
Notations

- 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: $n_i \in \mathbb{N}$ with $\sum_i n_i = W_{\text{total}}$.
  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 unit-message from P_1 to P_i: c.
  One-port bus: P_1 sends a *single* message at a time over the bus, all processors communicate at the same speed with the master.

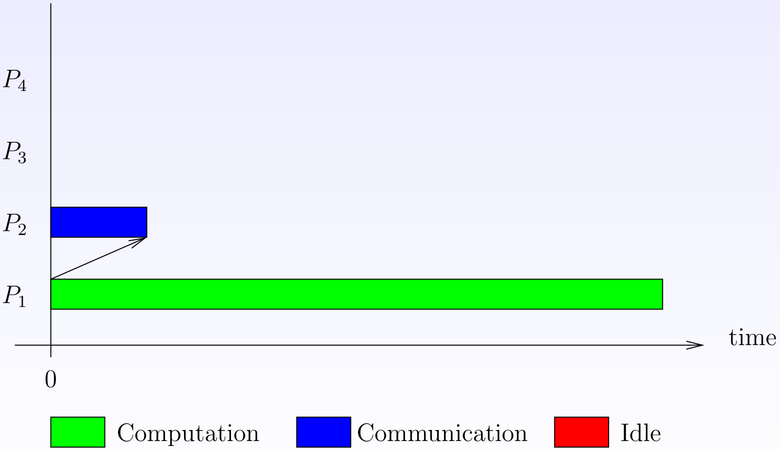
Behavior of the master and of the slaves (illustration)



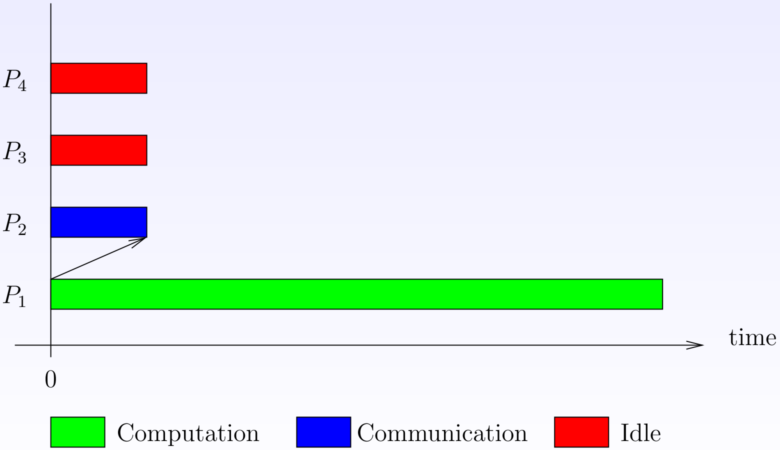
Behavior of the master and of the slaves (illustration)



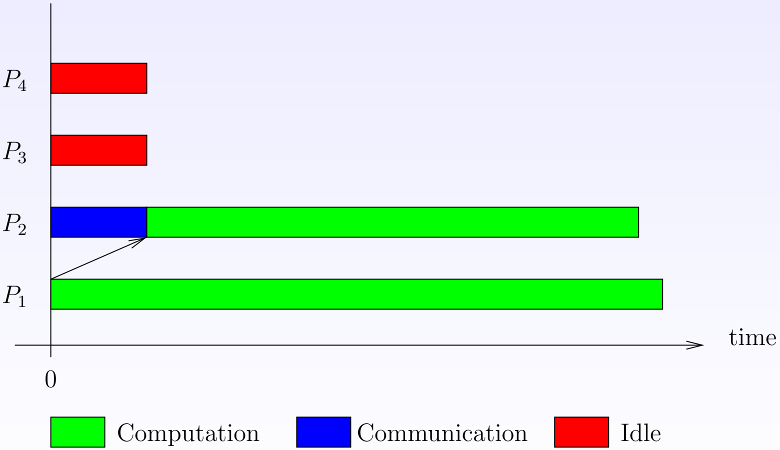
Behavior of the master and of the slaves (illustration)



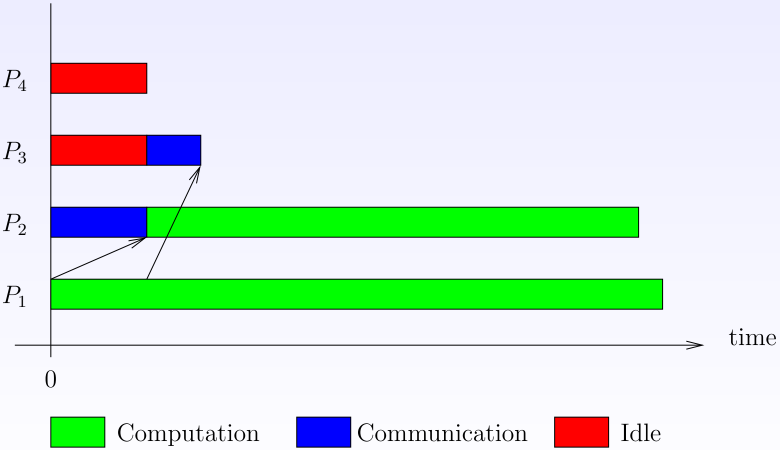
Behavior of the master and of the slaves (illustration)



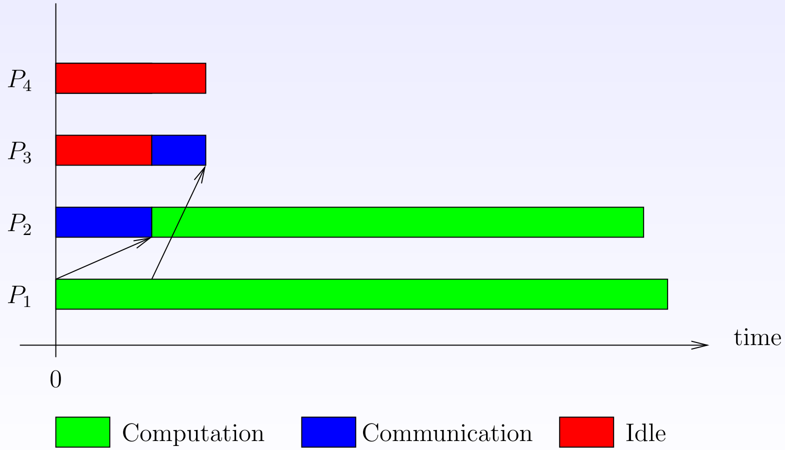
Behavior of the master and of the slaves (illustration)



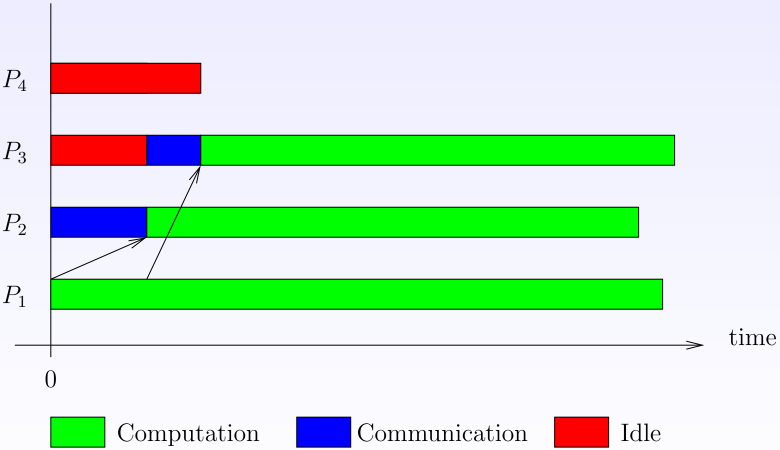
Behavior of the master and of the slaves (illustration)



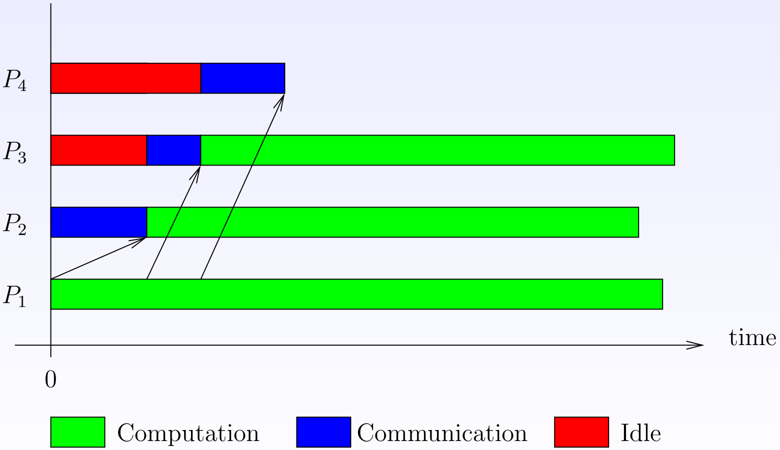
Behavior of the master and of the slaves (illustration)



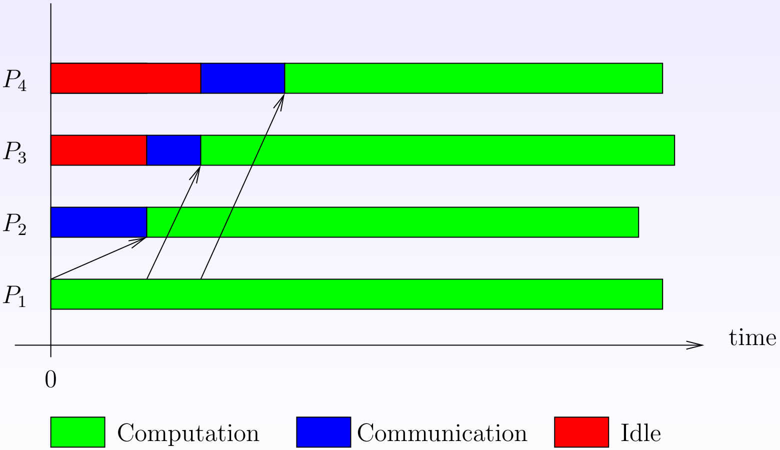
Behavior of the master and of the slaves (illustration)



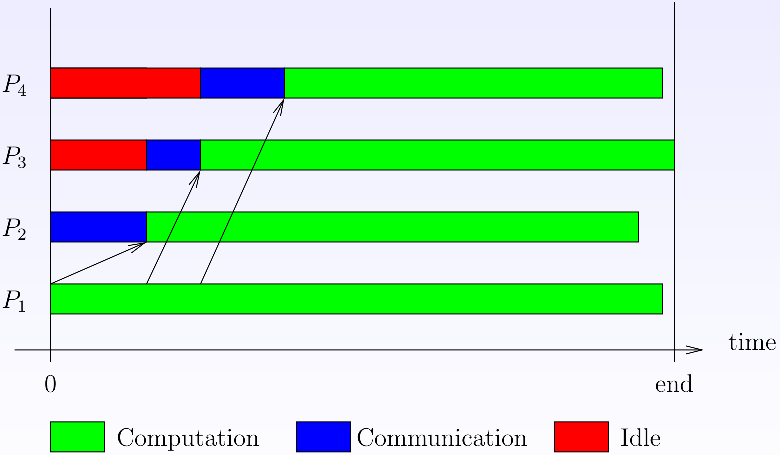
Behavior of the master and of the slaves (illustration)



Behavior of the master and of the slaves (illustration)



Behavior of the master and of the slaves (illustration)



Behavior of the master and of the slaves (hypotheses)

- The master sends its chunk of n_i data to processor P_i in a single sending.

Behavior of the master and of the slaves (hypotheses)

- The master sends its chunk of n_i data to processor P_i in a single sending.
- The master sends their data to the processors, serving one processor at a time, in the order P_2, ..., P_p.

Behavior of the master and of the slaves (hypotheses)

- The master sends its chunk of n_i data to processor P_i in a single sending.
- The master sends their data to the processors, serving one processor at a time, in the order P_2, ..., P_p.
- During this time the master processes its n_1 data.

Behavior of the master and of the slaves (hypotheses)

- The master sends its chunk of n_i data to processor P_i in a single sending.
- The master sends their data to the processors, serving one processor at a time, in the order P_2, ..., P_p.
- During this time the master processes its n_1 data.
- A slave does not start the processing of its data before it has received all of them.

Equations

- P_1: $T_1 = n_1.w_1$

Equations

- P_1: $T_1 = n_1.w_1$
- P_2: $T_2 = n_2.c + n_2.w_2$

Equations

- P_1: $T_1 = n_1.w_1$
- P_2: $T_2 = n_2.c + n_2.w_2$
- P_3: $T_3 = (n_2.c + n_3.c) + n_3.w_3$

Equations

- P_1: $T_1 = n_1.w_1$
- P_2: $T_2 = n_2.c + n_2.w_2$
- P_3: $T_3 = (n_2.c + n_3.c) + n_3.w_3$
- P_i: $T_i = \sum_{j=2}^{i} n_j.c + n_i.w_i$ for $i \geq 2$

Equations

- P_1: $T_1 = n_1.w_1$
- P_2: $T_2 = n_2.c + n_2.w_2$
- P_3: $T_3 = (n_2.c + n_3.c) + n_3.w_3$
- P_i: $T_i = \sum_{j=2}^{i} n_j.c + n_i.w_i$ for $i \geq 2$
- P_i: $T_i = \sum_{j=1}^{i} n_j.c_j + n_i.w_i$ for $i \geq 1$ with $c_1 = 0$ and $c_j = c$ otherwise.

Execution time

$$T = \max_{1 \leq i \leq p} \left(\sum_{j=1}^{i} n_j.c_j + n_i.w_i \right)$$

We look for a data distribution n_1, ..., n_p which minimizes T.

Execution time: rewriting

$$T = \max\left(n_1.c_1 + n_1.w_1, \max_{2 \leq i \leq p}\left(\sum_{j=1}^{i} n_j.c_j + n_i.w_i\right)\right)$$

$$T = n_1.c_1 + \max\left(n_1.w_1, \max_{2 \leq i \leq p}\left(\sum_{j=2}^{i} n_j.c_j + n_i.w_i\right)\right)$$

An optimal solution for the distribution of W_{total} data over p processors is obtained by distributing n_1 data to processor P_1 and then optimally distributing $W_{\text{total}} - n_1$ data over processors P_2 to P_p.

Algorithm

$\mathit{solution}[0, p] \leftarrow \mathrm{cons}(0, NIL)$; $\mathit{cost}[0, p] \leftarrow 0$
for $d \leftarrow 1$ **to** W_{total} **do**
  $\mathit{solution}[d, p] \leftarrow \mathrm{cons}(d, NIL)$
  $\mathit{cost}[d, p] \leftarrow d \cdot c_p + d \cdot w_p$
for $i \leftarrow p - 1$ **downto** 1 **do**
  $\mathit{solution}[0, i] \leftarrow \mathrm{cons}(0, \mathit{solution}[0, i + 1])$
  $\mathit{cost}[0, i] \leftarrow 0$
  for $d \leftarrow 1$ **to** W_{total} **do**
    $(sol, min) \leftarrow (0, \mathit{cost}[d, i + 1])$
    for $e \leftarrow 1$ **to** d **do**
      $m \leftarrow e \cdot c_i + \max(e \cdot w_i, \mathit{cost}[d - e, i + 1])$
      if $m < min$ **then**
        $(sol, min) \leftarrow (e, m)$
    $\mathit{solution}[d, i] \leftarrow \mathrm{cons}(sol, \mathit{solution}[d - sol, i + 1])$
    $\mathit{cost}[d, i] \leftarrow min$
return $(\mathit{solution}[W_{\text{total}}, 1], \mathit{cost}[W_{\text{total}}, 1])$

Complexity

- **Theorical complexity**

$$O(W_{\text{total}}^2 \cdot p)$$

- **Complexity in practice**
If $W_{\text{total}} = 817,101$ and $p = 16$, on a Pentium III running at 933 MHz: more than two days... (in 2002)
(Optimized version ran in 6 minutes.)

Disadvantages

- Cost
- Solution is not reusable
- Solution is only partial

We do not need the solution to be so precise

Overview

Notation

- 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 $\alpha_i W_{\text{total}} \in \mathbb{Q}$ and $\sum_i \alpha_i = 1$.**
  Length of a unit-size work on processor P_i: w_i.
  Computation time on P_i: $\alpha_i W_{\text{total}} w_i$.
- Time needed to send a unit-message from P_1 to P_i: c.
  One-port model: P_1 sends a *single* message at a time, all processors communicate at the same speed with the master.

Equations

For processor P_i (with $c_1 = 0$ and $c_j = c$ otherwise):

$$T_i = \sum_{j=1}^{i} \alpha_j W_{\mathsf{total}}.c_j + \alpha_i W_{\mathsf{total}}.w_i$$

$$T = \max_{1 \leq i \leq p} \left(\sum_{j=1}^{i} \alpha_j W_{\mathsf{total}}.c_j + \alpha_i W_{\mathsf{total}}.w_i \right)$$

We look for a data distribution α_1, ..., α_p which minimizes T.

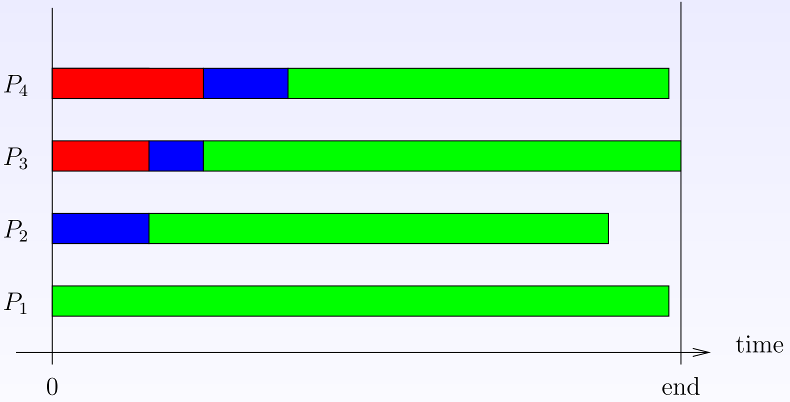
Properties of load-balancing

Lemma

In an optimal solution, all processors end their processing at the same time.

Demonstration of lemma 1

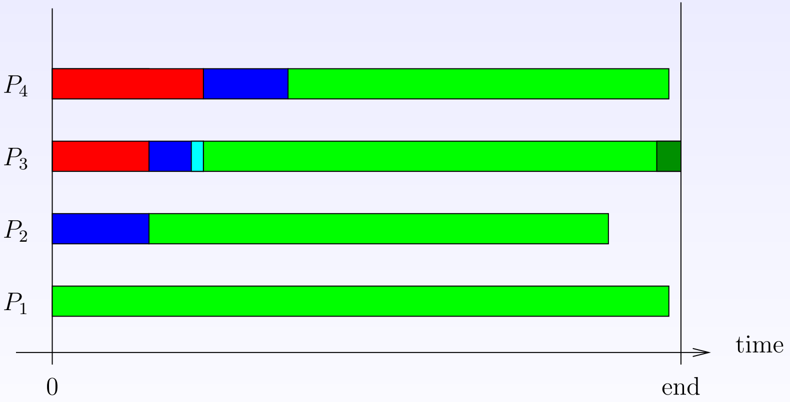
Two slaves i and $i+1$ with $T_i < T_{i+1}$.



We decrease α_{i+1} by ϵ.

Demonstration of lemma 1

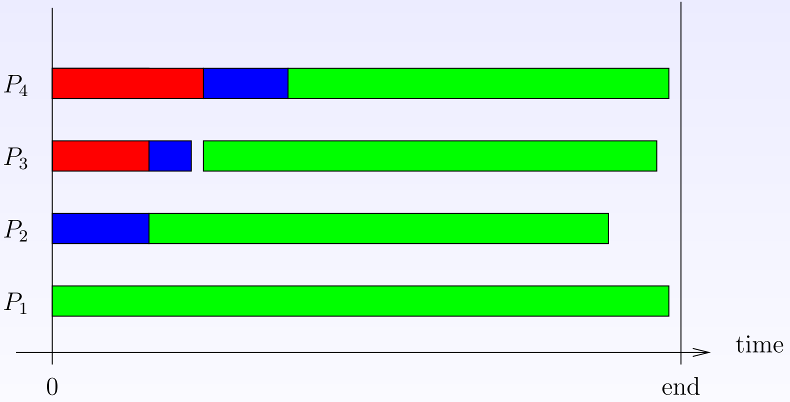
Two slaves i and $i+1$ with $T_i < T_{i+1}$.



We decrease α_{i+1} by ϵ.

Demonstration of lemma 1

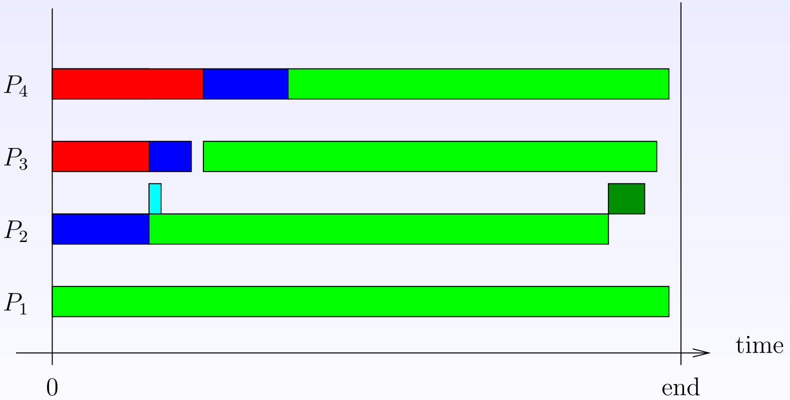
Two slaves i and $i+1$ with $T_i < T_{i+1}$.



We decrease α_{i+1} by ϵ.

Demonstration of lemma 1

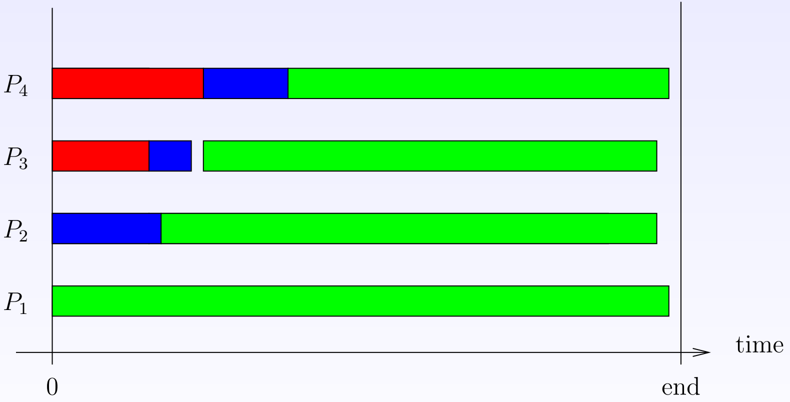
Two slaves i and $i+1$ with $T_i < T_{i+1}$.



We increase α_i by ϵ.

Demonstration of lemma 1

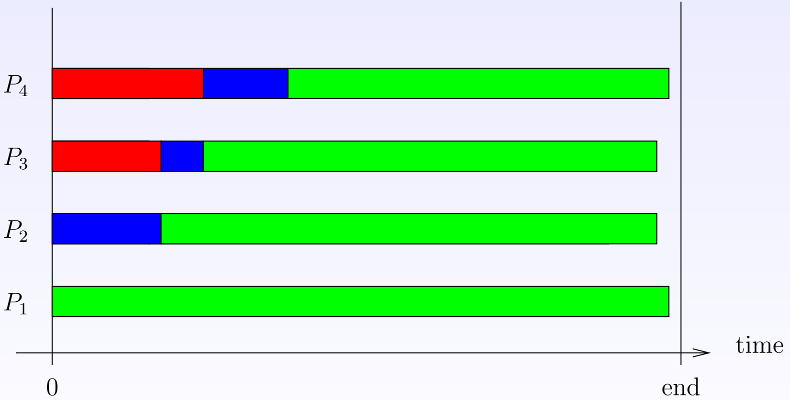
Two slaves i and $i+1$ with $T_i < T_{i+1}$.



We increase α_i by ϵ.

Demonstration of lemma 1

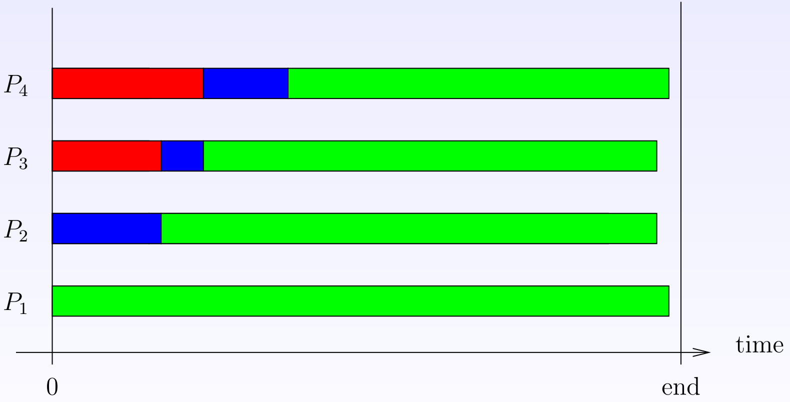
Two slaves i and $i+1$ with $T_i < T_{i+1}$.



The communication time for the following processors is unchanged.

Demonstration of lemma 1

Two slaves i and $i+1$ with $T_i < T_{i+1}$.



We end up with a better solution!

Demonstration of lemma 1 (continuation and conclusion)

- Ideal: $T'_i = T'_{i+1}$.
  We choose ϵ such that:

$$(\alpha_i + \epsilon) W_{\mathsf{total}} (c + w_i) = (\alpha_i + \epsilon) W_{\mathsf{total}} c + (\alpha_{i+1} - \epsilon) W_{\mathsf{total}} (c + w_{i+1})$$

- The master stops before the slaves: absurd.
- The master stops after the slaves: we decrease P_1 by ϵ.

Property for the selection of ressources

Lemma

In an optimal solution all processors work.

Property for the selection of ressources

Lemma

In an optimal solution all processors work.

Demonstration: this is just a corollary of lemma 1...

Resolution

$$T = \alpha_1 W_{\mathsf{total}} w_1.$$

Resolution

$T = \alpha_1 W_{\text{total}} w_1$.

$T = \alpha_2 (c + w_2) W_{\text{total}}$. Therefore $\alpha_2 = \frac{w_1}{c+w_2}\alpha_1$.

Resolution

$T = \alpha_1 W_{\mathsf{total}} w_1$.

$T = \alpha_2 (c + w_2) W_{\mathsf{total}}$. Therefore $\alpha_2 = \frac{w_1}{c+w_2}\alpha_1$.

$T = (\alpha_2 c + \alpha_3 (c + w_3)) W_{\mathsf{total}}$. Therefore $\alpha_3 = \frac{w_2}{c+w_3}\alpha_2$.

Resolution

$T = \alpha_1 W_{\text{total}} w_1$.

$T = \alpha_2(c + w_2)W_{\text{total}}$. Therefore $\alpha_2 = \frac{w_1}{c+w_2}\alpha_1$.

$T = (\alpha_2 c + \alpha_3(c + w_3))W_{\text{total}}$. Therefore $\alpha_3 = \frac{w_2}{c+w_3}\alpha_2$.

$\alpha_i = \frac{w_{i-1}}{c+w_i}\alpha_{i-1}$ for $i \geq 2$.

Resolution

$T = \alpha_1 W_{\mathsf{total}} w_1$.

$T = \alpha_2 (c + w_2) W_{\mathsf{total}}$. Therefore $\alpha_2 = \frac{w_1}{c+w_2}\alpha_1$.

$T = (\alpha_2 c + \alpha_3 (c + w_3)) W_{\mathsf{total}}$. Therefore $\alpha_3 = \frac{w_2}{c+w_3}\alpha_2$.

$\alpha_i = \frac{w_{i-1}}{c+w_i}\alpha_{i-1}$ for $i \geq 2$.

$\sum_{i=1}^{n} \alpha_i = 1$.

Resolution

$T = \alpha_1 W_{\text{total}} w_1$.

$T = \alpha_2 (c + w_2) W_{\text{total}}$. Therefore $\alpha_2 = \frac{w_1}{c+w_2}\alpha_1$.

$T = (\alpha_2 c + \alpha_3 (c + w_3)) W_{\text{total}}$. Therefore $\alpha_3 = \frac{w_2}{c+w_3}\alpha_2$.

$\alpha_i = \frac{w_{i-1}}{c+w_i}\alpha_{i-1}$ for $i \geq 2$.

$\sum_{i=1}^{n} \alpha_i = 1$.

$$\alpha_1 \left(1 + \frac{w_1}{c + w_2} + \ldots + \prod_{k=2}^{j} \frac{w_{k-1}}{c + w_k} + \ldots \right) = 1$$

Impact of the order of communications

How important is the influence of the ordering of the processor on the solution ?

?

No impact of the order of the communications

Volume processed by processors P_i and P_{i+1} during a time T.

No impact of the order of the communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i**:** $\alpha_i(c + w_i)W_{\text{total}} = T$. Therefore $\alpha_i = \frac{1}{c+w_i}\frac{T}{W_{\text{total}}}$.

No impact of the order of the communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i**:** $\alpha_i(c + w_i)W_{\text{total}} = T$. Therefore $\alpha_i = \frac{1}{c+w_i}\frac{T}{W_{\text{total}}}$.

Processor P_{i+1}**:** $\alpha_i c W_{\text{total}} + \alpha_{i+1}(c + w_{i+1})W_{\text{total}} = T$.

No impact of the order of the communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i**:** $\alpha_i(c + w_i)W_{\text{total}} = T$. Therefore $\alpha_i = \frac{1}{c+w_i}\frac{T}{W_{\text{total}}}$.

Processor P_{i+1}**:** $\alpha_i c W_{\text{total}} + \alpha_{i+1}(c + w_{i+1})W_{\text{total}} = T$.
Thus $\alpha_{i+1} = \frac{1}{c+w_{i+1}}(\frac{T}{W_{\text{total}}} - \alpha_i c) = \frac{w_i}{(c+w_i)(c+w_{i+1})}\frac{T}{W_{\text{total}}}$.

No impact of the order of the communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i**:** $\alpha_i(c + w_i)W_{\text{total}} = T$. Therefore $\alpha_i = \frac{1}{c+w_i}\frac{T}{W_{\text{total}}}$.

Processor P_{i+1}**:** $\alpha_i c W_{\text{total}} + \alpha_{i+1}(c + w_{i+1})W_{\text{total}} = T$.
Thus $\alpha_{i+1} = \frac{1}{c+w_{i+1}}(\frac{T}{W_{\text{total}}} - \alpha_i c) = \frac{w_i}{(c+w_i)(c+w_{i+1})}\frac{T}{W_{\text{total}}}$.

Processors P_i **and** P_{i+1}**:**

$$\alpha_i + \alpha_{i+1} = \frac{c + w_i + w_{i+1}}{(c + w_i)(c + w_{i+1})}\frac{T}{W_{\text{total}}}$$

Choice of the master processor

We compare processors P_1 and P_2.

Choice of the master processor

We compare processors P_1 and P_2.

Processor P_1**:** $\alpha_1 w_1 W_{\text{total}} = T$. Then, $\alpha_1 = \frac{1}{w_1} \frac{T}{W_{\text{total}}}$.

Choice of the master processor

We compare processors P_1 and P_2.

Processor P_1**:** $\alpha_1 w_1 W_{\text{total}} = T$. Then, $\alpha_1 = \frac{1}{w_1} \frac{T}{W_{\text{total}}}$.

Processor P_2**:** $\alpha_2 (c + w_2) W_{\text{total}} = T$. Thus, $\alpha_2 = \frac{1}{c+w_2} \frac{T}{W_{\text{total}}}$.

Choice of the master processor

We compare processors P_1 and P_2.

Processor P_1: $\alpha_1 w_1 W_{\text{total}} = T$. Then, $\alpha_1 = \frac{1}{w_1} \frac{T}{W_{\text{total}}}$.

Processor P_2: $\alpha_2 (c + w_2) W_{\text{total}} = T$. Thus, $\alpha_2 = \frac{1}{c+w_2} \frac{T}{W_{\text{total}}}$.

Total volume processed:

$$\alpha_1 + \alpha_2 = \frac{c + w_1 + w_2}{w_1(c + w_2)} = \frac{c + w_1 + w_2}{c w_1 + w_1 w_2}$$

Choice of the master processor

We compare processors P_1 and P_2.

Processor P_1: $\alpha_1 w_1 W_{\text{total}} = T$. Then, $\alpha_1 = \frac{1}{w_1}\frac{T}{W_{\text{total}}}$.

Processor P_2: $\alpha_2 (c + w_2) W_{\text{total}} = T$. Thus, $\alpha_2 = \frac{1}{c+w_2}\frac{T}{W_{\text{total}}}$.

Total volume processed:

$$\alpha_1 + \alpha_2 = \frac{c + w_1 + w_2}{w_1(c + w_2)} = \frac{c + w_1 + w_2}{cw_1 + w_1 w_2}$$

Minimal when $w_1 < w_2$.
Master = the most powerfull processor (for computations).

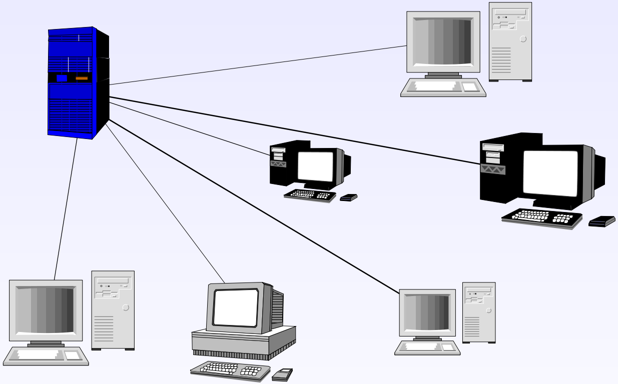
Conclusion

- Closed-form expressions for the execution time and the distribution of data.
- Choice of the master.
- The ordering of the processors has no impact.
- All processors take part in the work.

Overview

1. The context
2. Bus-like network: classical resolution
3. Bus-like network: resolution under the divisible load model
4. Star-like network
5. Multi-round algorithms
6. Conclusion

Star-like network



- The links between the master and the slaves have *different* characteristics.
- The slaves have different computational power.

Notation

- 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 unit-message from P_1 to P_i: c_i.**
  One-port model: P_1 sends a *single* message at a time.

Impact of the order of communications

?

Impact of the order of communications

Volume processed by processors P_i and P_{i+1} during a time T.

Impact of the order of communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i: $\alpha_i(c_i + w_i)W_{\text{total}} = T$. Thus, $\alpha_i = \frac{1}{c_i+w_i}\frac{T}{W_{\text{total}}}$.

Impact of the order of communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i**:** $\alpha_i(c_i + w_i)W_{\text{total}} = T$. Thus, $\alpha_i = \frac{1}{c_i+w_i}\frac{T}{W_{\text{total}}}$.

Processor P_{i+1}**:** $\alpha_i c_i W_{\text{total}} + \alpha_{i+1}(c_{i+1} + w_{i+1})W_{\text{total}} = T$.

Impact of the order of communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i: $\alpha_i(c_i + w_i)W_{\text{total}} = T$. Thus, $\alpha_i = \frac{1}{c_i+w_i}\frac{T}{W_{\text{total}}}$.

Processor P_{i+1}: $\alpha_i c_i W_{\text{total}} + \alpha_{i+1}(c_{i+1} + w_{i+1})W_{\text{total}} = T$.
Thus, $\alpha_{i+1} = \frac{1}{c_{i+1}+w_{i+1}}(1 - \frac{c_i}{c_i+w_i})\frac{T}{W_{\text{total}}} = \frac{w_i}{(c_i+w_i)(c_{i+1}+w_{i+1})}\frac{T}{W_{\text{total}}}$.

Impact of the order of communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i**:** $\alpha_i(c_i + w_i)W_{\text{total}} = T$. Thus, $\alpha_i = \frac{1}{c_i+w_i}\frac{T}{W_{\text{total}}}$.

Processor P_{i+1}**:** $\alpha_i c_i W_{\text{total}} + \alpha_{i+1}(c_{i+1} + w_{i+1})W_{\text{total}} = T$.
Thus, $\alpha_{i+1} = \frac{1}{c_{i+1}+w_{i+1}}(1 - \frac{c_i}{c_i+w_i})\frac{T}{W_{\text{total}}} = \frac{w_i}{(c_i+w_i)(c_{i+1}+w_{i+1})}\frac{T}{W_{\text{total}}}$.

Volume processed: $\alpha_i + \alpha_{i+1} = \frac{c_{i+1}+w_i+w_{i+1}}{(c_i+w_i)(c_{i+1}+w_{i+1})}$

Impact of the order of communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i: $\alpha_i(c_i + w_i)W_{\text{total}} = T$. Thus, $\alpha_i = \frac{1}{c_i+w_i}\frac{T}{W_{\text{total}}}$.

Processor P_{i+1}: $\alpha_i c_i W_{\text{total}} + \alpha_{i+1}(c_{i+1} + w_{i+1})W_{\text{total}} = T$.
Thus, $\alpha_{i+1} = \frac{1}{c_{i+1}+w_{i+1}}(1 - \frac{c_i}{c_i+w_i})\frac{T}{W_{\text{total}}} = \frac{w_i}{(c_i+w_i)(c_{i+1}+w_{i+1})}\frac{T}{W_{\text{total}}}$.

Volume processed: $\alpha_i + \alpha_{i+1} = \frac{c_{i+1}+w_i+w_{i+1}}{(c_i+w_i)(c_{i+1}+w_{i+1})}$

Communication time: $\alpha_i c_i + \alpha_{i+1}c_{i+1} = \frac{c_i c_{i+1}+c_{i+1}w_i+c_i w_{i+1}}{(c_i+w_i)(c_{i+1}+w_{i+1})}$

Impact of the order of communications

Volume processed by processors P_i and P_{i+1} during a time T.

Processor P_i**:** $\alpha_i(c_i + w_i)W_{\text{total}} = T$. Thus, $\alpha_i = \frac{1}{c_i+w_i}\frac{T}{W_{\text{total}}}$.

Processor P_{i+1}**:** $\alpha_i c_i W_{\text{total}} + \alpha_{i+1}(c_{i+1} + w_{i+1})W_{\text{total}} = T$.
Thus, $\alpha_{i+1} = \frac{1}{c_{i+1}+w_{i+1}}(1 - \frac{c_i}{c_i+w_i})\frac{T}{W_{\text{total}}} = \frac{w_i}{(c_i+w_i)(c_{i+1}+w_{i+1})}\frac{T}{W_{\text{total}}}$.

Volume processed: $\alpha_i + \alpha_{i+1} = \frac{c_{i+1}+w_i+w_{i+1}}{(c_i+w_i)(c_{i+1}+w_{i+1})}$

Communication time: $\alpha_i c_i + \alpha_{i+1}c_{i+1} = \frac{c_i c_{i+1}+c_{i+1}w_i+c_i w_{i+1}}{(c_i+w_i)(c_{i+1}+w_{i+1})}$

Processors must be served by decreasing bandwidths.

Ressource selection

Lemma

In an optimal solution, all processors work.

Demonstration of lemma 3

We take an optimal solution. Let P_k be a processor which does not receive any work: we put it last in the processor ordering and we give it a fraction α_k such that $\alpha_k(c_k + w_k)W_{\text{total}}$ is equal to the processing time of the last processor which received some work.

Demonstration of lemma 3

We take an optimal solution. Let P_k be a processor which does not receive any work: we put it last in the processor ordering and we give it a fraction α_k such that $\alpha_k(c_k + w_k)W_{\text{total}}$ is equal to the processing time of the last processor which received some work.

Why should we put this processor last ?

Load-balancing property

Lemma

In an optimal solution, all processors end at the same time.

Demonstration of lemma 4

- Most existing proofs are false.

MINIMIZE T,
SUBJECT TO

$$\begin{cases} & \sum_{i=1}^{n} \alpha_i \geq 1 \\ \forall i, & \alpha_i \geq 0 \\ \forall i, & \sum_{k=1}^{i} \alpha_k c_k + \alpha_i w_i \leq T \end{cases}$$

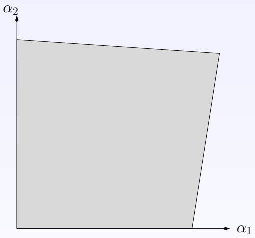
Demonstration of lemma 4

- Most existing proofs are false.

MINIMIZE T,

SUBJECT TO

$$\begin{cases} & \sum_{i=1}^{n} \alpha_i \geq 1 \\ \forall i, & \alpha_i \geq 0 \\ \forall i, & \sum_{k=1}^{i} \alpha_k c_k + \alpha_i w_i \leq T \end{cases}$$



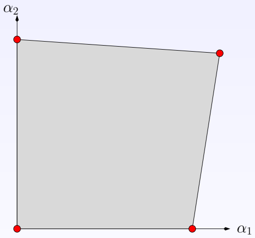
Demonstration of lemma 4

- Most existing proofs are false.

MINIMIZE T,

SUBJECT TO

$$\begin{cases} & \sum_{i=1}^{n} \alpha_i \geq 1 \\ \forall i, & \alpha_i \geq 0 \\ \forall i, & \sum_{k=1}^{i} \alpha_k c_k + \alpha_i w_i \leq T \end{cases}$$



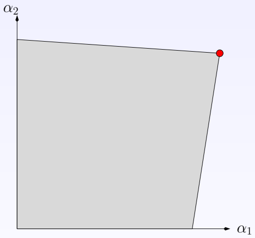
Demonstration of lemma 4

- Most existing proofs are false.

MINIMIZE T,

SUBJECT TO

$$\begin{cases} & \sum_{i=1}^{n} \alpha_i \geq 1 \\ \forall i, & \alpha_i \geq 0 \\ \forall i, & \sum_{k=1}^{i} \alpha_k c_k + \alpha_i w_i \leq T \end{cases}$$



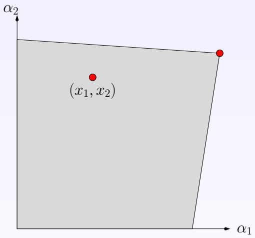
Demonstration of lemma 4

- Most existing proofs are false.

MINIMIZE T,

SUBJECT TO

$$\begin{cases} & \sum_{i=1}^{n} \alpha_i \geq 1 \\ \forall i, & \alpha_i \geq 0 \\ \forall i, & \sum_{k=1}^{i} \alpha_k c_k + \alpha_i w_i \leq T \end{cases}$$



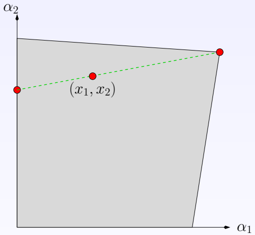
Demonstration of lemma 4

- Most existing proofs are false.

MINIMIZE T,

SUBJECT TO

$$\begin{cases} & \sum_{i=1}^{n} \alpha_i \geq 1 \\ \forall i, & \alpha_i \geq 0 \\ \forall i, & \sum_{k=1}^{i} \alpha_k c_k + \alpha_i w_i \leq T \end{cases}$$

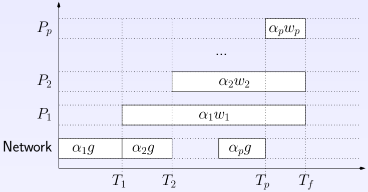


Conclusion

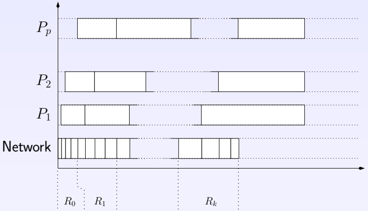
- The processors must be ordered by decreasing bandwidths
- All processors are working
- All processors end their work at the same time
- Formulas for the execution time and the distribution of data

Overview

One round vs. multi-round

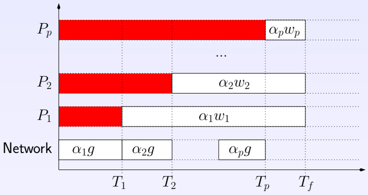


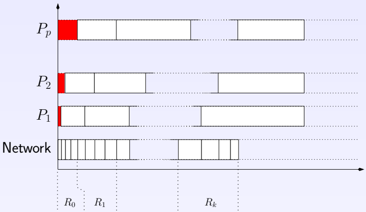
One round



Multi-round

One round vs. multi-round





One round

$\rightsquigarrow$ long idle-times

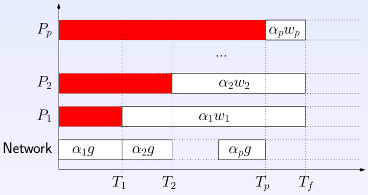
Multi-round

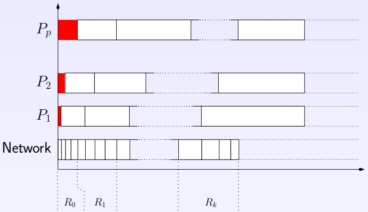
Efficient when W_{total} large

Intuition: start with small rounds, then increase chunks.

Problems:

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 chunks.

Problems:

- linear communication model leads to absurd solution
- resource selection
- number of rounds
- size of each round

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

Definition (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 $T \ge 0$, 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.

Complexity

Definition (One round, $\forall i, c_i = 0$)

Given W_{total}, p workers, $(P_i)_{1 \leq i \leq p}$, $(L_i)_{1 \leq i \leq p}$, 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).

Fixed activation sequence

Hypotheses

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

MINIMIZE T, UNDER THE CONSTRAINTS

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

Can be solved in polynomial time.

Fixed number of activations

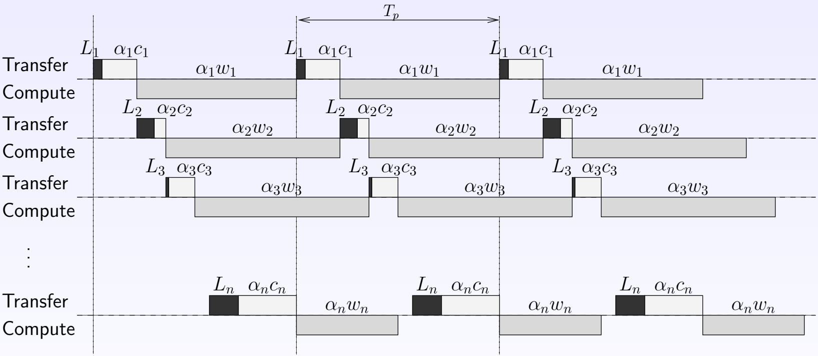
MINIMIZE T, UNDER THE CONSTRAINTS

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

Exact but exponential

Can lead to branch-and-bound algorithms

Periodic schedule



How to choose T_p? Which resources to select?

With no overlap (1/4)

Equations

- Divide total execution time T into k periods of duration T_p.

With no overlap (1/4)

Equations

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

With no overlap (1/4)

Equations

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

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

With no overlap (1/4)

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:

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

With no overlap (2/4)

Normalization

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

With no overlap (2/4)

Normalization

- β_i average number of tasks processed by P_i during one time unit.
- Linear program: $\begin{array}{l} \text{MAXIMIZE} \sum_{i=1}^{p} \beta_i \\ \begin{cases} \forall i \in \mathcal{I}, \quad \beta_i(c_i + w_i) \leq 1 - \frac{L_i}{T_p} \\ \sum_{i \in \mathcal{I}} \beta_i c_i \leq 1 - \frac{\sum_{i \in \mathcal{I}} L_i}{T_p} \end{cases} \end{array}$.

With no overlap (2/4)

Normalization

- β_i average number of tasks processed by P_i during one time unit.
- Linear program: $$\left\{ \begin{array}{ll} \multicolumn{2}{l}{\textsc{Maximize} \sum_{i=1}^{p} \beta_i} \\ \forall i \in \mathcal{I}, & \beta_i(c_i + w_i) \leq 1 - \frac{L_i}{T_p} \\ \sum_{i \in \mathcal{I}} \beta_i c_i \leq 1 - \frac{\sum_{i \in \mathcal{I}} L_i}{T_p} & \end{array} \right. .$$

Relaxed version

$$\left\{ \begin{array}{ll} \multicolumn{2}{l}{\textsc{Maximize} \sum_{i=1}^{p} x_i} \\ \forall 1 \leq i \leq p, & x_i(c_i + w_i) \leq 1 - \frac{L_i}{T_p} \\ \sum_{i=1}^{p} x_i c_i \leq 1 - \frac{\sum_{i=1}^{p} L_i}{T_p} & \end{array} \right.$$

With no overlap (2/4)

Normalization

- β_i average number of tasks processed by P_i during one time unit.
- Linear program: $$\left\{\begin{array}{ll} \multicolumn{2}{l}{\textsc{Maximize} \sum_{i=1}^{p} \beta_i} \\ \forall i \in \mathcal{I}, & \beta_i(c_i + w_i) \leq 1 - \frac{L_i}{T_p} \\ \sum_{i \in \mathcal{I}} \beta_i c_i \leq 1 - \frac{\sum_{i \in \mathcal{I}} L_i}{T_p} & \end{array}\right. .$$

Relaxed version

$$\left\{\begin{array}{ll} \multicolumn{2}{l}{\textsc{Maximize} \sum_{i=1}^{p} x_i} \\ \forall 1 \leq i \leq p, & x_i(c_i + w_i) \leq 1 - \frac{\sum_{i=1}^{p} L_i}{T_p} \\ \sum_{i=1}^{p} x_i c_i \leq 1 - \frac{\sum_{i=1}^{p} L_i}{T_p} & \end{array}\right.$$

With no overlap (3/4)

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

With no overlap (4/4)

Asymptotic optimality

- Let $T_p = \sqrt{T^*_{\max}}$ and $\alpha_i = x_i T_p$ for all i.

With no overlap (4/4)

Asymptotic optimality

- Let $T_p = \sqrt{T^*_{\max}}$ and $\alpha_i = x_i T_p$ for all i.
- Then $T \leq T^*_{\max} + O(\sqrt{T^*_{\max}})$.

With no overlap (4/4)

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.

Overview

What should be remembered?

- Underlying principle: we may not need the optimal solution; approximated solutions may be as good and far easier to achieve
- Communications costs may play a far bigger role in designing solutions than computation costs