Divisible load theory

Loris Marchal

November 5, 2012

1 The context

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

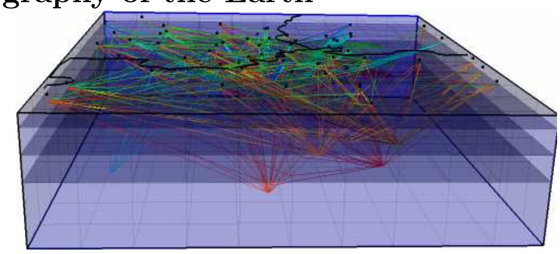
Execution platforms : Distributed heterogeneous platforms (network of workstations, clusters, clusters of clusters, grids, etc.)

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 : 817101
– 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 loads model

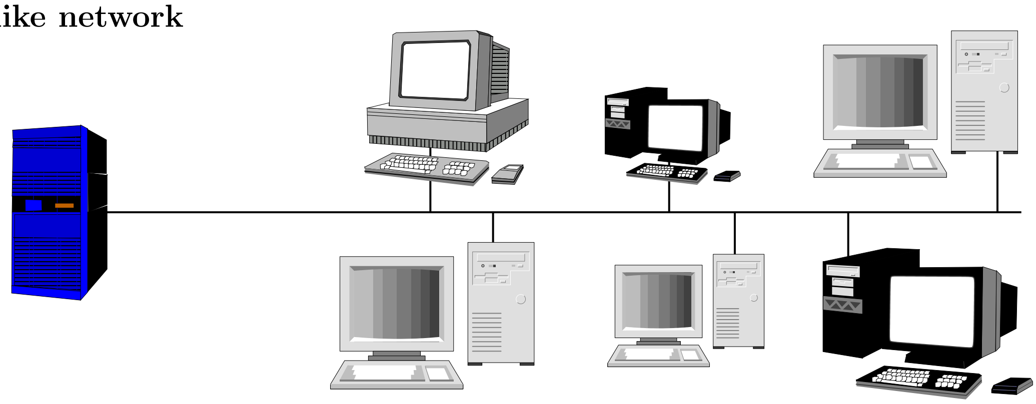
Applications made of a very (very) large number of fine grain computations.

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

Independent computations : neither synchronizations nor communications.

2 Bus-like network : classical resolution

Bus-like network

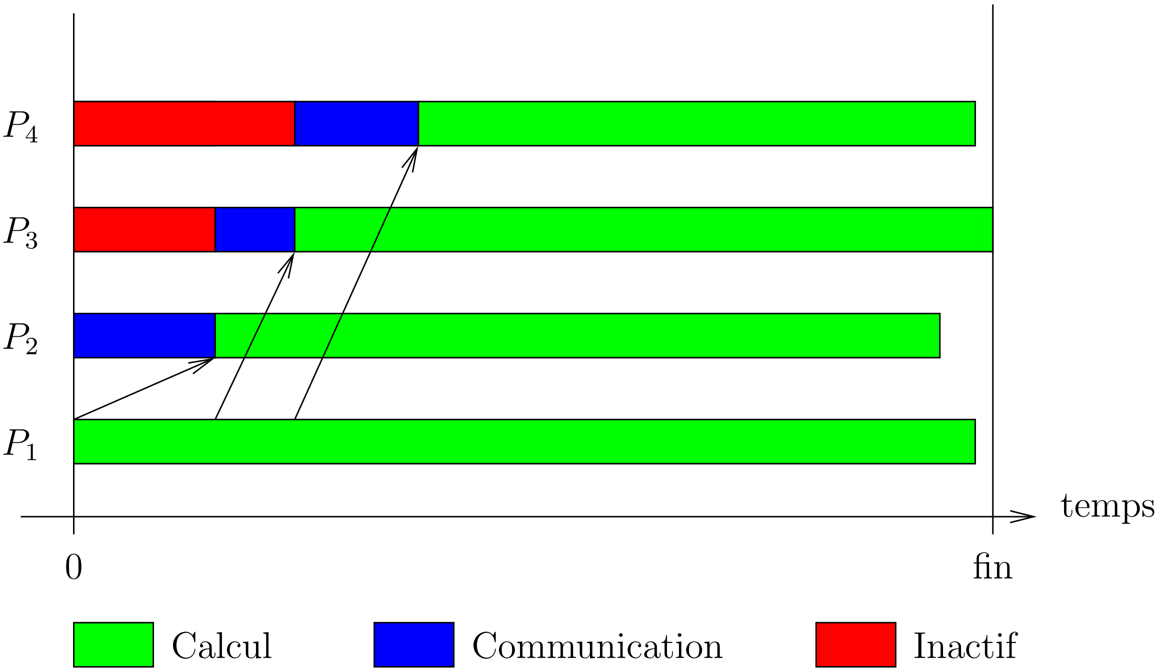


– The links between the master and the slaves all have the same characteristics.
– The slave have different computation power.

Notations

– A set P_1, ..., P_p of processors
– 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 (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

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

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

Complexity

- **Theoretical complexity**

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

- **Complexity in practice**
  If $W_{\text{total}} = 817101$ and $p = 16$, on a Pentium III running at 933 MHz : more than two days...
  (Optimized version ran in 6 minutes.)

Disadvantages

- Cost
- Solution is not reusable
- Solution is only partial (processor order is fixed)

We do not need the solution to be so precise

3 Bus-like network : resolution under the divisible load 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 $\alpha_i W_{\textbf{total}}$ with $\alpha_i W_{\textbf{total}} \in \mathbb{Q}$ and $\sum_i \alpha_i = 1$.**
  Length of a unit-size work on processor P_i : w_i.
  Computation time on P_i : $\alpha_i w_i$.
- 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) :

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

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

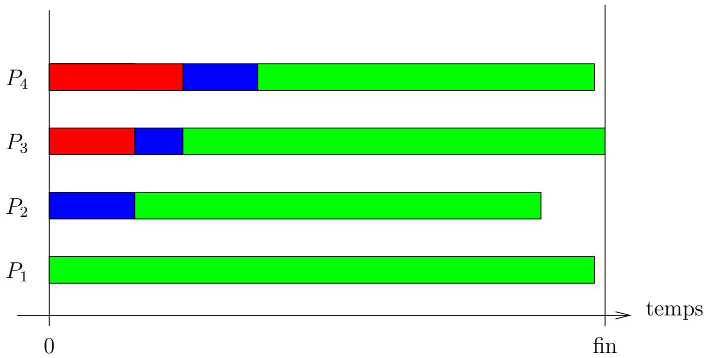
We look for a data distribution α_1, ..., α_p which minimizes T.

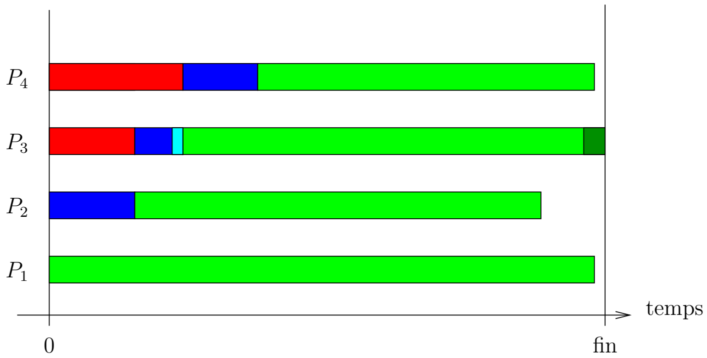
Properties of load-balancing

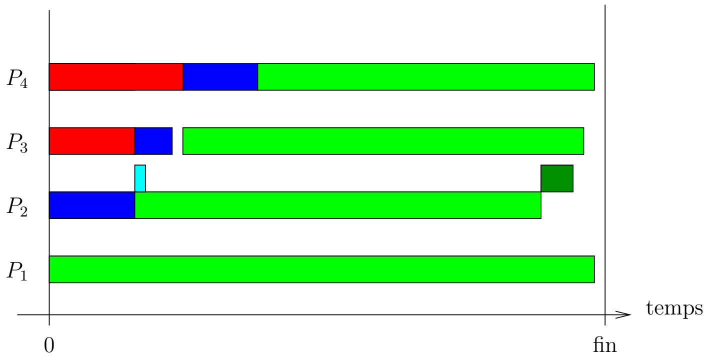
Lemma: In an optimal solution, all processors end their processing at the same time.

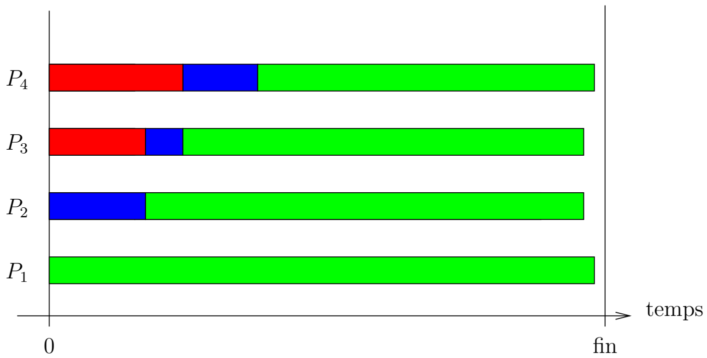
Demonstration of lemma 1

Two slaves i and $i+1$ with $T_i < T_{i+1}$.
(the same results holds with $i-1$ and i)









We decrease α_{i+1} by ϵ.We increase α_i by ϵ.The communication time for the following processors is unchanged.We end up with a better solution !

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

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

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

Property for the selection of resources

Lemma: In an optimal solution all processors work.

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

Resolution for a given ordering of the workers

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

$T = \alpha_2 (c + w_2) W_{\text{total}}$. 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 $i \geq 2$.

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

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

Impact of the order of communications

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

Consider the 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} :

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

No impact of the order of the communications

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 powerful processor (for computations).

Conclusion

– Closed-form expressions for the execution time and the distribution of data.

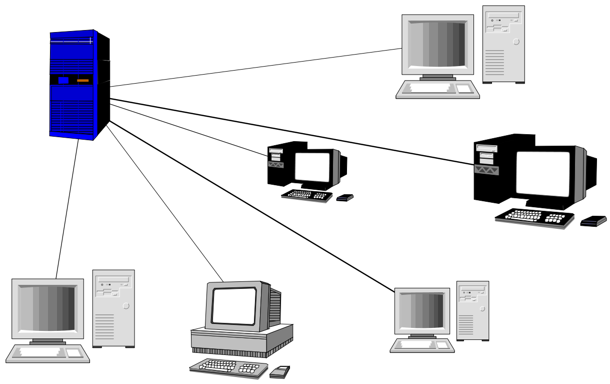
– Choice of the master.

– The ordering of the processors has no impact.

– All processors take part in the work.

4 Star-like network

Star-like network



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

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 $\alpha_i W_{\text{total}} \in \mathbb{Q}$ and $\sum_i \alpha_i = 1$.
– Length of a unit-size work on processor P_i : w_i.
– Computation time on P_i : $\alpha_i w_i$.
– **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.

Resource selection

Lemma: In an optimal solution, all processors work.

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.

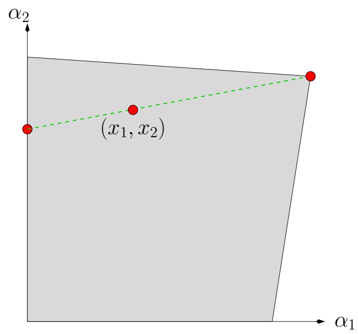
Why should we put this processor last ?

Load-balancing property

Lemma: In an optimal solution, all processors end at the same time.
– Most existing proofs are false.

MINIMIZE T,
SUBJECT TO

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



– The constraints define a polyhedron
– One of the optimal solution is a vertex of the polyhedron, that is at least n among the $2n$ inequalities are equalities,
– It can not be a lower bound, because all processors participate, thus the only vertex with all $\alpha_i > 0$ is an optimal solution
– Assume there is another optimal solution ; it lies within the polyhedron
– We can build a linear combination of both optimal solution (with optimal objective), such that one variable is zero, which contradicts the previous result.

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.

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

5 Conclusion

What should we remember from that ?

- A simple, approximate, model is sometimes better than a complete, but intractable one.
- In such a configuration (bus network, one-port model), communication times are more important than computation speed.
- Always start with a very simple model, (solve it) and then make it more complex if possible.