Length-based Blocking and Local Estimations in Distributed Simulation: A Case Study

C.D. Pham S. Fdida

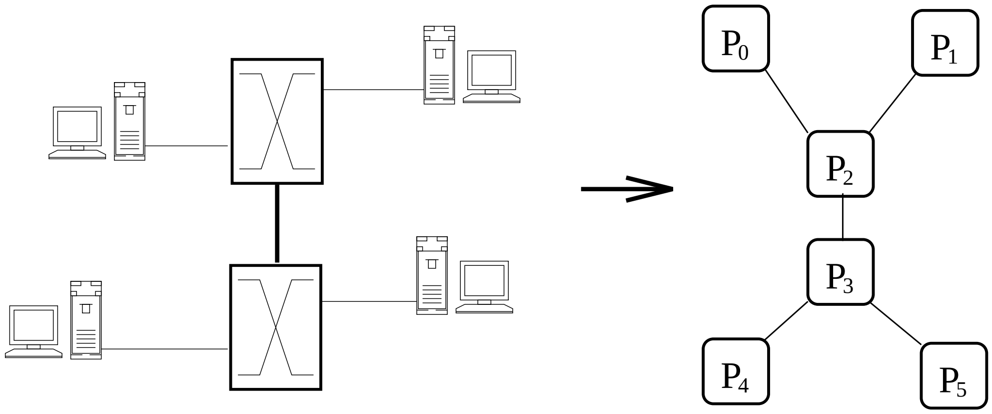
Laboratoire MASI
Université Pierre et Marie Curie
4 place Jussieu 75252 Paris Cedex 05
{pham,fdida}@masi.ibp.fr

April 10, 1996
29th Annual Simulation Symposium

Laboratoire Masi

Parallel Simulation and The Process Synchronization Problem

Logical Process modeling and Spacial partionning



Event lists and virtual clocks are distributed.

⬇

Causality is required to obtain correct results.

⬇

A synchronization algorithm is needed to respect causality.

⬇

These synchronization mechanisms fall in two categories: **conservative** and **optimistic**.

Synchronization Mechanisms

Conservative Approach

- avoid all cases of time errors by blocking LPs
- null-messages to avoid deadlock,
- lookahead ability is essential.

Optimistic Approach

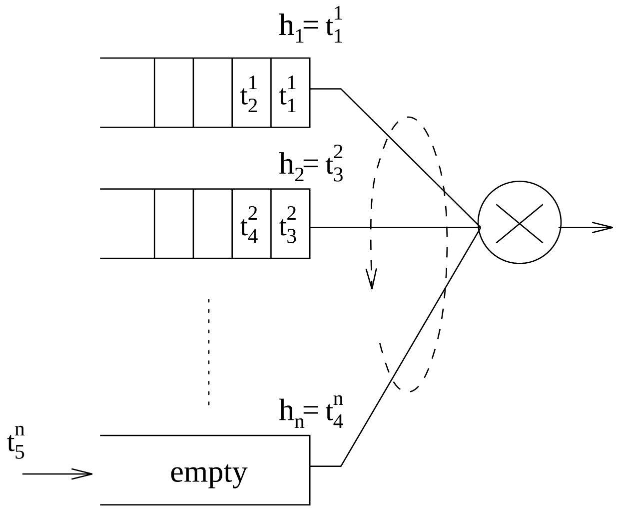
- rollback mechanism,
- periodic state saving,
- anti-messages.

☞ More complex than a conservative approach.

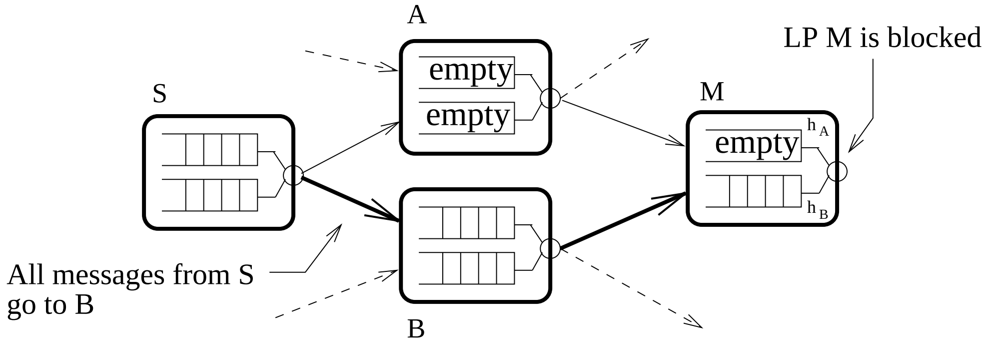
Conservative Approaches

Conservative algorithms (Chandy-Misra-Bryant) avoid all cases of time errors by blocking LPs.

The basic conservative algorithm

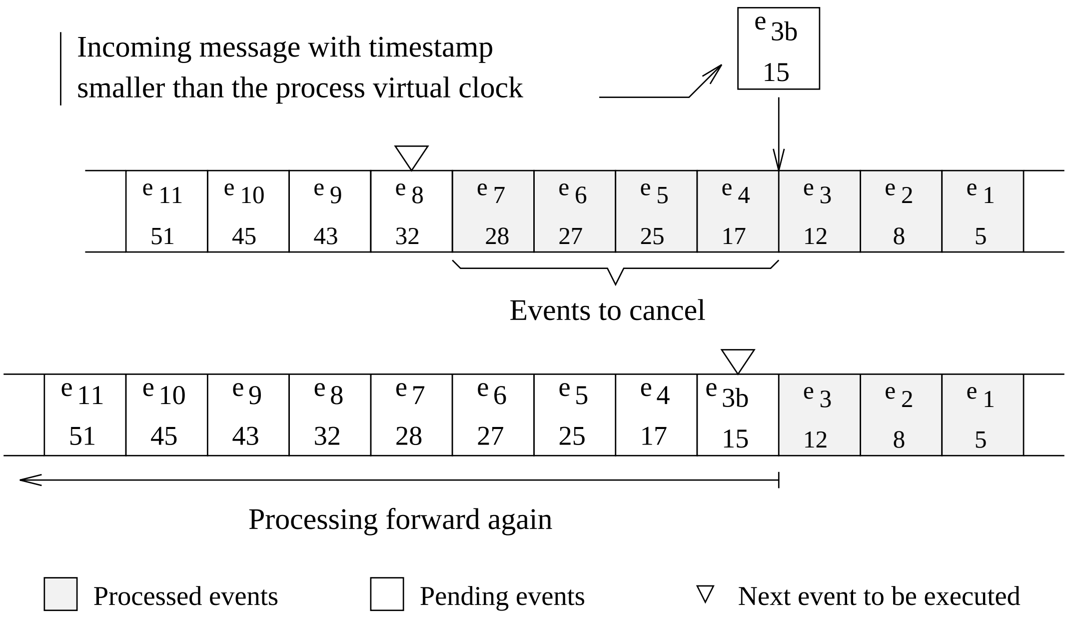


Deadlock $\rightarrow$ null-messages and lookahead ability



Optimistic Approaches

Optimistic algorithms (Time Warp) implement a rollback mechanism and allow an LP to process all available input messages.



Need of periodic state saving and anti-messages to:

- Restore the process state,
- Send anti-messages to other processes,
- Re-execute events in the right order.

Local estimations

Optimistic scheduling is always optimal if the overhead of the rollback mechanism could be greatly reduced.

⬇

- ☐ Local estimations are used as a substitute to rollback.
- ☐ Some errors are accepted in order to achieve better speedup.
- ☐ Mean value statistics should be preserved correctly.

➡ Inefficient if the number of time errors is too high.

☞ Reduce the number of time errors

Blocking strategies

- ☐ Blocking: a way to decrease the time error probability.
- ☐ Two classes of strategies: time-based and length-based

Time-Based

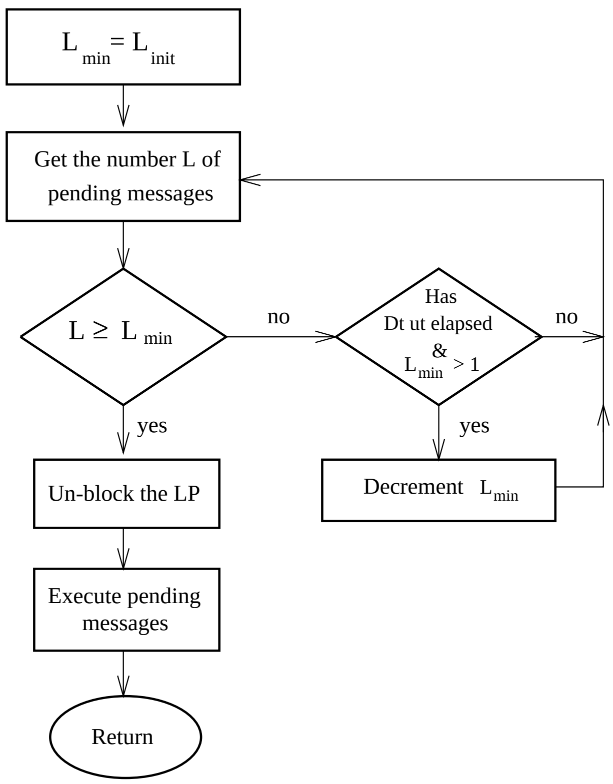
- continous tuning.
- the load imbalance problem is lessened.

Length-Based

- more natural way to block.
- the blocking overhead can be reduced.

The length-based blocking strategy

- Preventive strategy based on the length of the input queue.
- A threshold L_{min} is defined.
  - ➥ Deadlock can occur.
  - ➥ L_{min} is decreased every D_t ut.



Flow chart of the blocking algorithm

Setting L_{init} and D_t

2 SOLUTIONS

L_{init} small and D_t large

- if messages arrive rapidly
  - length condition rapidly satisfied.
- if messages arrive slowly
  - lot of time wasted.

L_{init} large and D_t small

- if messages arrive rapidly
  - length condition rapidly satisfied.
- if messages arrive slowly
  - L_{init} decreases rapidly.

☞ The second solution is more adaptive.

Computation of local estimations

- Let L_s be the list of saved states
- Let $S_i = (o_1^i, o_2^i, last_1^i, last_2^i, n_r^i, n_s^i)$ be the ith saved state. A saved state S_i has a timestamp T_i.

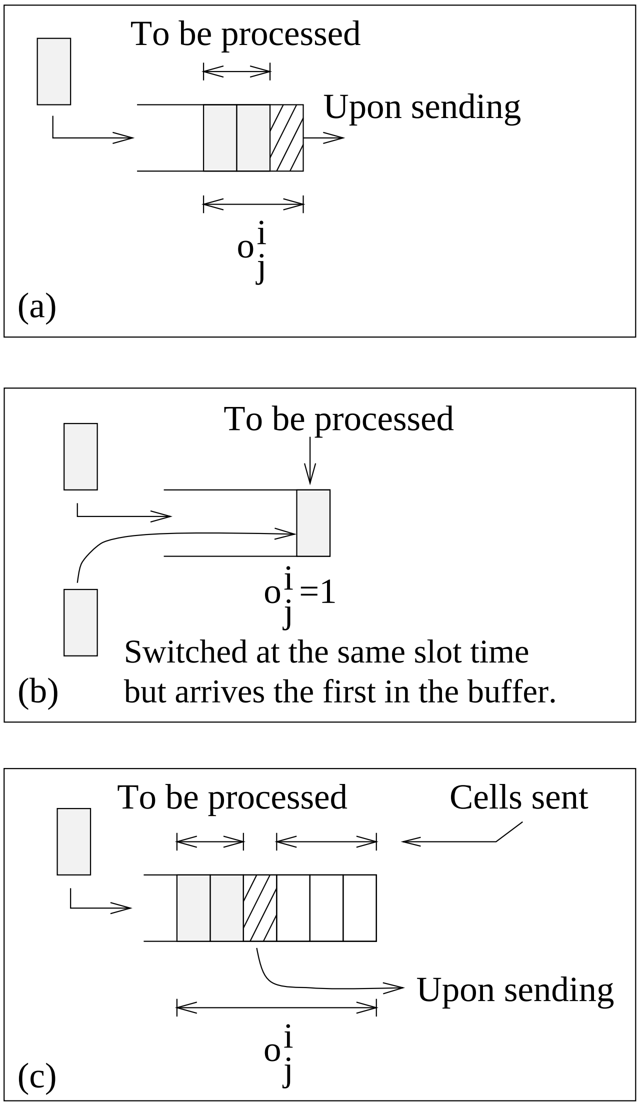
Estimation algorithm at the switch process

⬇

1. Gets the first state before time t from L_s.
2. Determines the output j, derives the date of departure t_d of this new cell by computing its waiting time and schedules a $< SEND, t_d >$ event for itself.
3. Inserts the new state in the list just before S_i.
4. Re-evaluates all states S_k with $k > i + 1$.

Computation of t_d

- Step 2 $\begin{cases} \text{if } t = last_j^i \text{ then } t_w = max\{o_j^i - 1, 1\} \\ \text{ else } t_w = max\{o_j^i - 1 - (t - T_i), 0\} \\ t_d = t + t_w + 1 + 1 \end{cases}$



New state and state re-evaluation

New process state S_{i+1}

- Step 3 $\left\{\begin{array}{l} o_j^{i+1} = max\{o_j^i - (t - T_i), 0\} + 1 \\ last_j^{i+1} = t \\ n_r^{i+1} = n_r^i + 1 \\ n_s^{i+1} = n_s^i + min\{o_j^i, t - T_i\} \end{array}\right.$

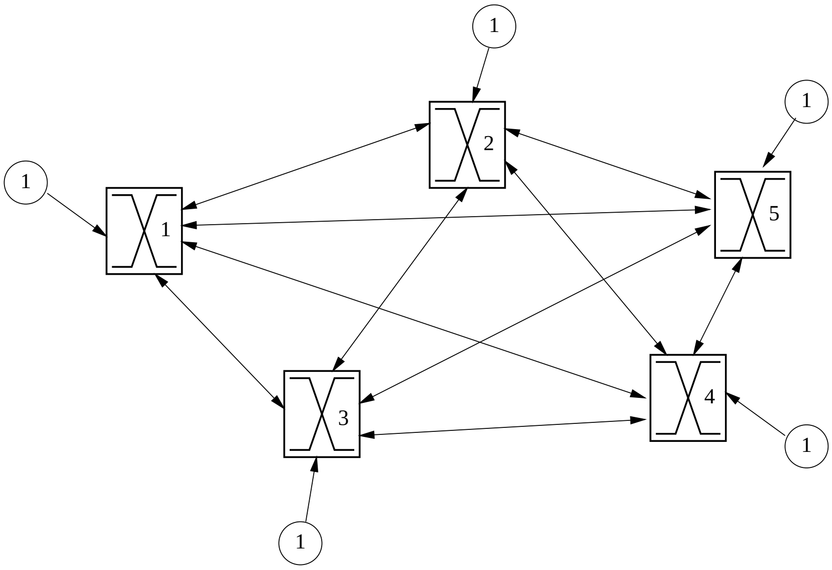
Re-evaluation of all states S_k with $k > i + 1$

- Step 4 $\left\{\begin{array}{l} \text{if } T_k - t < o_j^{i+1} \text{ then } o_j^k = o_j^k + 1 \\ \quad \text{else } o_j^k \text{ is unchanged} \\ last_j^k \text{ is unchanged} \\ n_r^k = n_r^k + 1 \\ \text{if } t_d < T_k \text{ then } n_s^k = n_s^k + 1 \\ \quad \text{else } n_s^k \text{ is unchanged} \end{array}\right.$

Test cases

FULLY CONNECTED NETWORK

SIMULATION AT THE CELL LEVEL

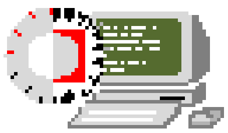


Network configuration for $N_s = 5$

- ☞ Locality is high
- ☞ Computational grain is low

Test environment

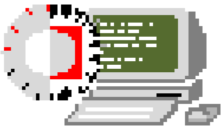
2 CONFIGURATIONS ARE USED.

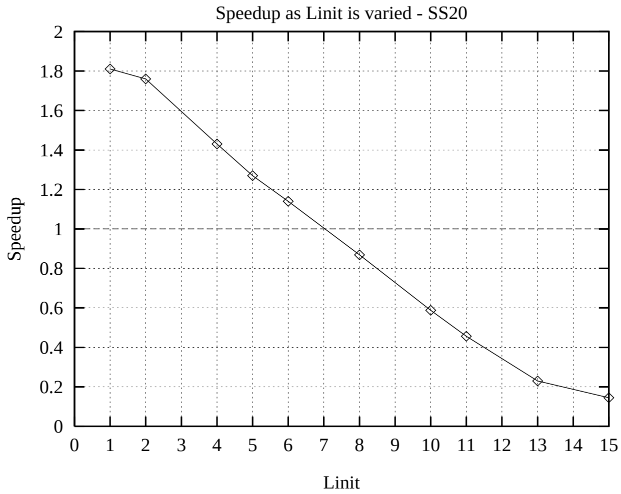


SS20 with 64MB memory - PVM library

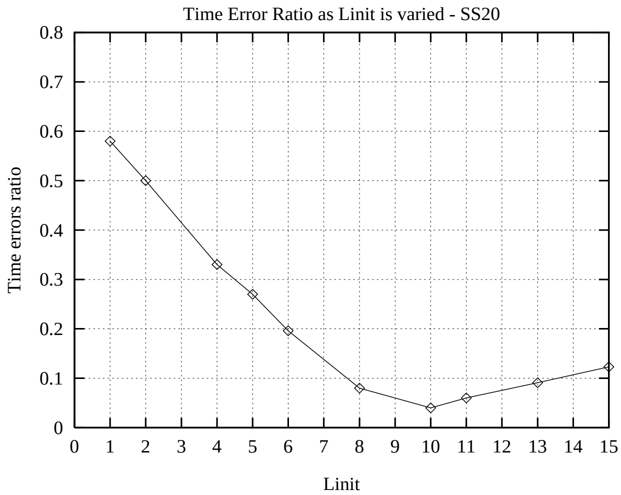
CM-5 with 32 processors - CM/PVM library

Empirical optimal value for L_{init}



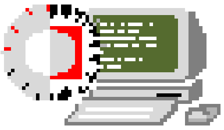


Speedup as L_{init} is varied



TER as L_{init} is varied

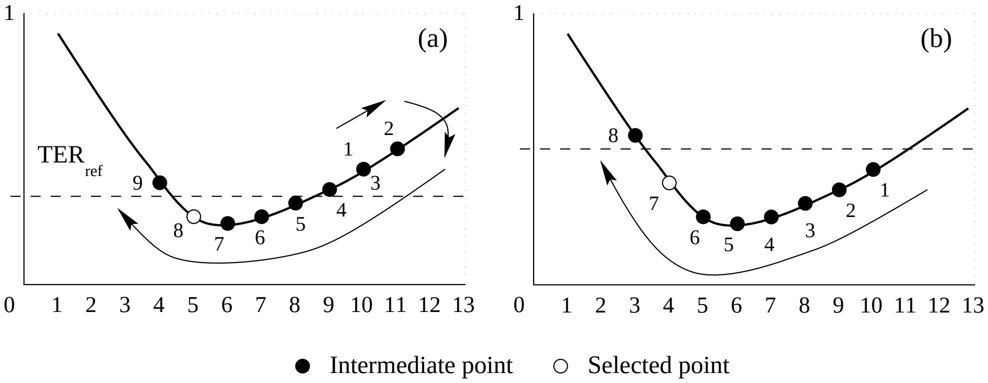
Percentage of error as L_{init} is varied



L_{init}	MBL	% of error	TER
15	3.48	+5.51%	0.123
13	3.40	+3.03%	0.091
11	3.45	+4.54%	0.060
CMB	3.30	0%	0
10	3.20	-2.75%	0.044
8	3.19	-3.10%	0.080
6	3.15	-4.55%	0.196
5	2.93	-10.97%	0.270
4	2.85	-13.52%	0.330
2	2.65	-19.70%	0.500
1	2.52	-23.40%	0.580

% of error and TER as L_{init} is varied

Automatic tuning of L_{init}

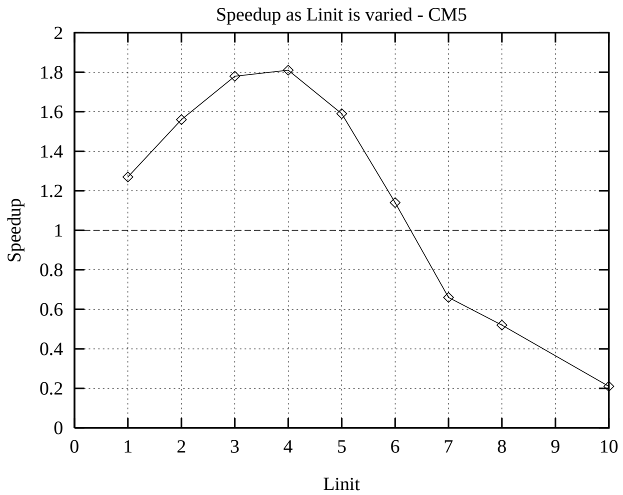


Initial point on the bad branch.

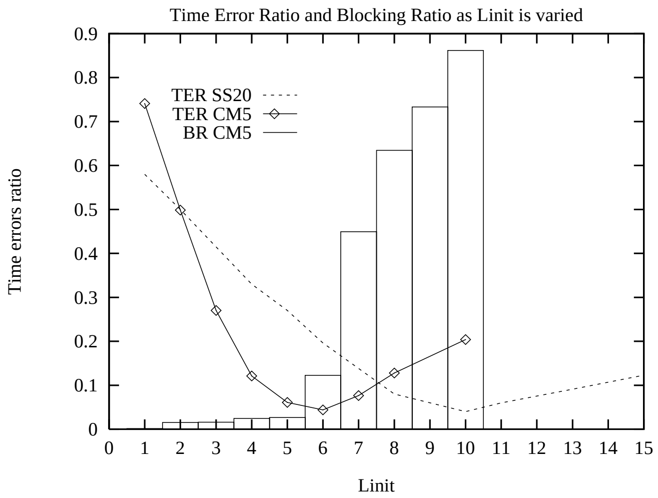
TER_{ref}	TER_{comp}	MBL	% of error	L_{init}
0.044	0.044	1.27	+1.56%	6
CMB	0	1.25	0%	—
0.080	0.061	1.22	-1.69%	5
0.196	0.121	1.20	-3.72%	4
0.270	0.268	1.08	-13.6%	3
0.330	0.270	1.07	-13.6%	3
0.500	0.498	0.93	-25.6%	2
0.580	0.496	0.93	-25.6%	2
1.0	0.740	0.77	-33.88%	1

L_{init} as TER_{ref} is varied

Speedup, TER and Blocking Ratio



Speedup as L_{init} is varied



TER and BR as L_{init} is varied

Varying the network size and the message density

preliminary results

Increasing the network size

- The optimal value for L_{init} does not increase linearly.
- The TER is reduced more quickly.

☞ Tuning can be difficult.

Increasing the message density

- Performances of conservative methods are likely to be improved.
- If the network's size is also large, the speedup can be greater than 1.8.

Conclusion and Perspectives

- ✓ A simple length-based blocking strategy has been proposed.
- ✓ Local estimations work better to give more accurate results.
- ✓ Simulation with ad-hoc estimations can give an error less than 5% for a speedup of 1.8.
- ✓ L_{init} should not be chosen too large.
- ✓ The length-based blocking strategy performs better when messages arrive rapidly.
- ✓ The desired level of accuracy could only be met approximately.
- ✗ Varying the network size and the message density.
- ✗ Comparaison with a time-based blocking strategy.