Load Balanced Parallel Simulated Annealing on a Cluster of SMP Nodes

Agnieszka Debudaj-Grabysz[1] and Rolf Rabenseifner[2]

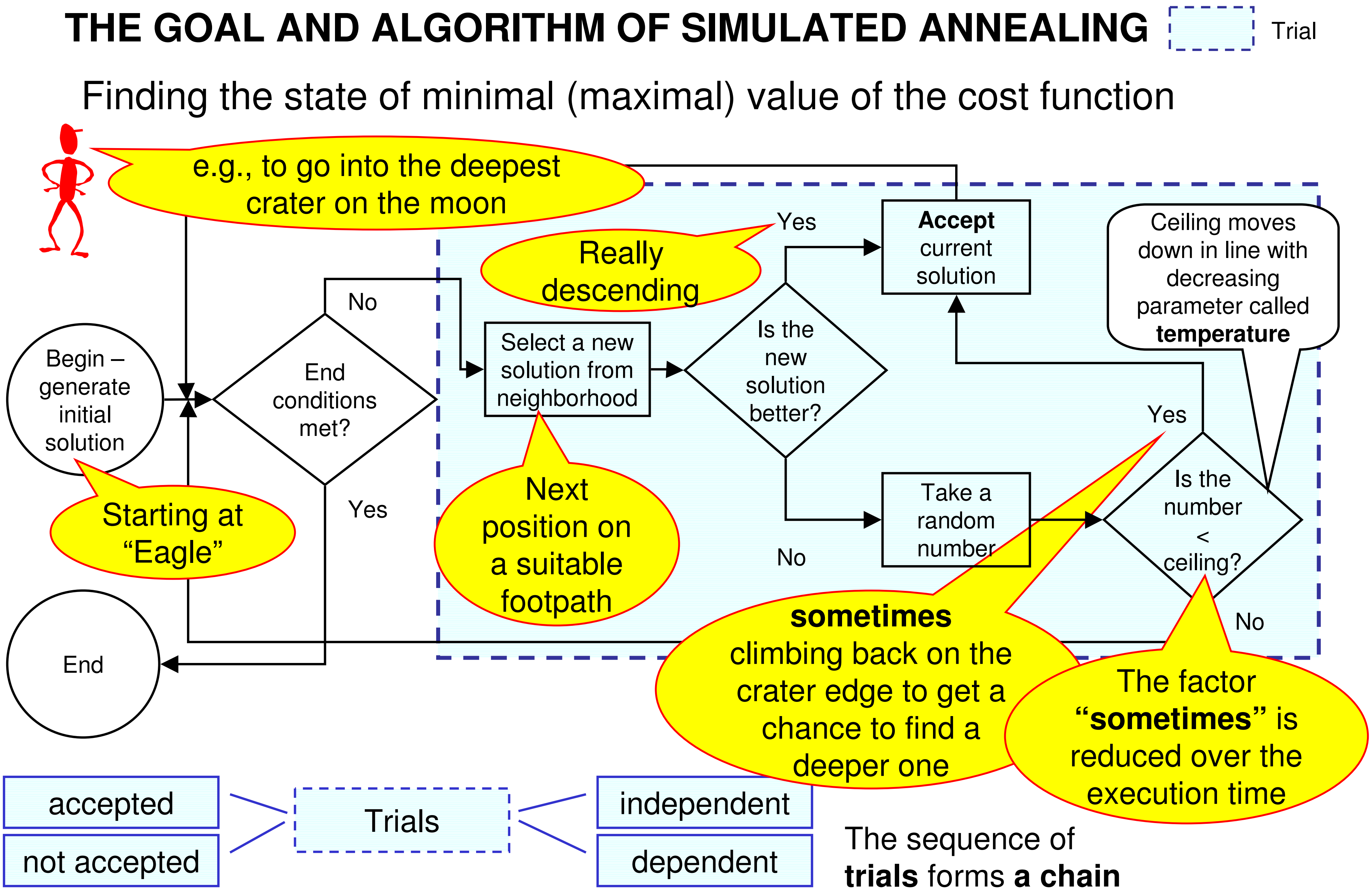
[1]Silesian University of Technology, Gliwice, Poland;
[2]High-Performance Computing Center (HLRS), Stuttgart, Germany

OUTLINE

1. The algorithm of simulated annealing
2. Hybrid communication method (HC) – nesting OpenMP in MPI
   - The reference method
   - The method with a single data exchange
4. Outer level load balancing
5. Inner level load balancing
6. Vehicle routing problem with time windows (VRPTW) – an example of a bi-criterion optimization problem
7. Experimental results

THE GOAL AND ALGORITHM OF SIMULATED ANNEALING

Finding the state of minimal (maximal) value of the cost function

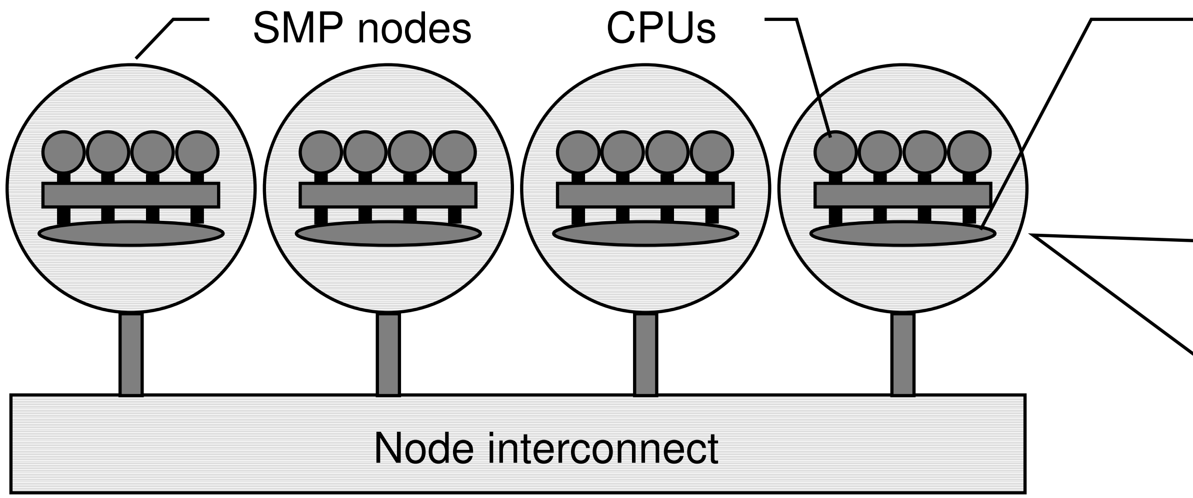


The sequence of **trials** forms **a chain**

THE HYBRID COMMUNICATION METHOD – Nesting OpenMP in MPI

Hybrid character of the hardware

Parallelization uses two levels



INNER PARALLELIZATION

- Only up to a few threads are used
- Communication overhead is minimal relative to that between nodes
- Uses OpenMP

OUTER PARALLELIZATION

- Several SMP nodes
- Communication overhead is not negligible
- Uses MPI

THE REFERENCE HYBRID COMMUNICATION METHOD

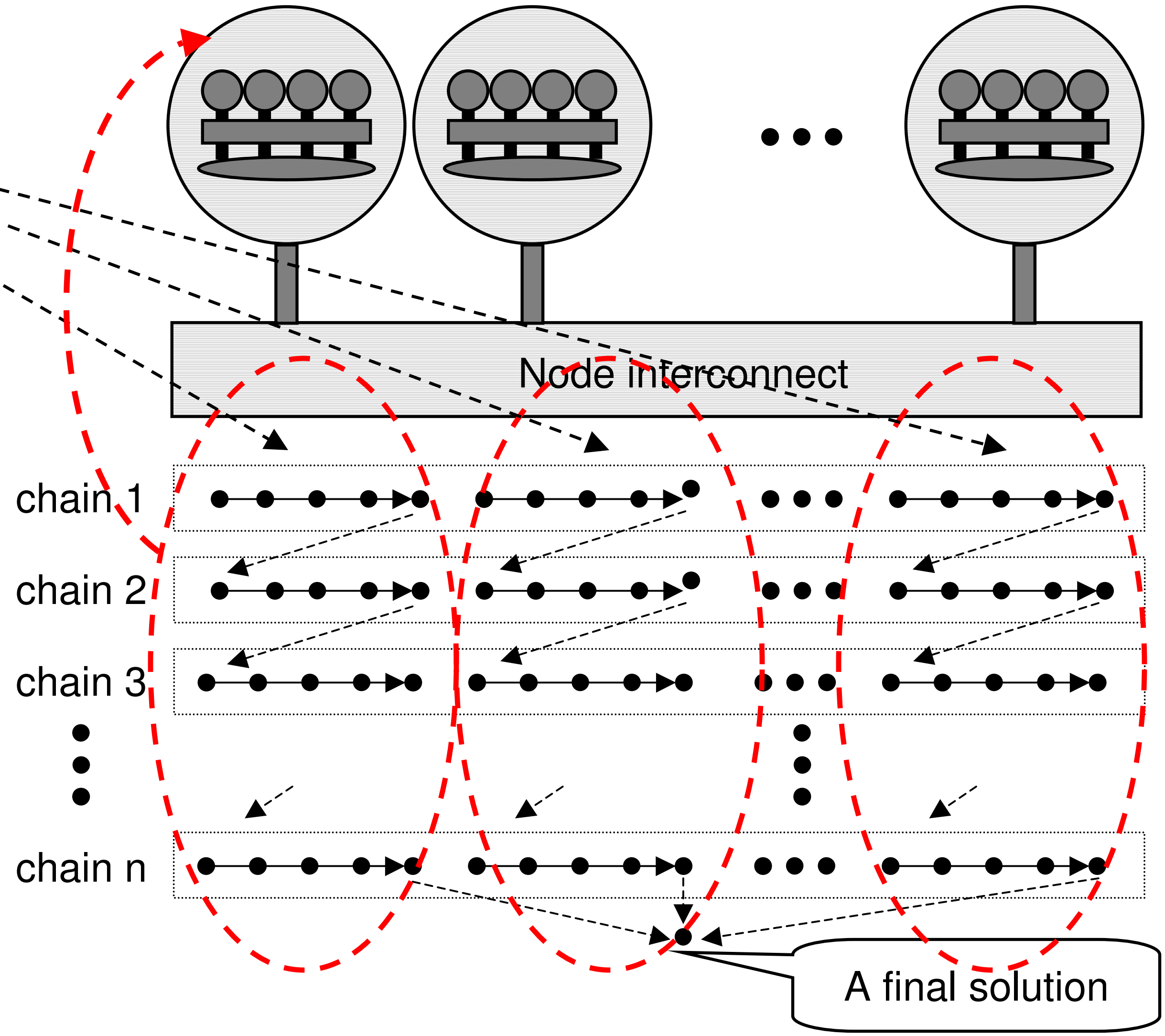
Outer parallelization for communication between nodes

Each chain is divided into sub-chains by the factor of the number of nodes

The process of generating every consecutive chain is performed without communication between nodes

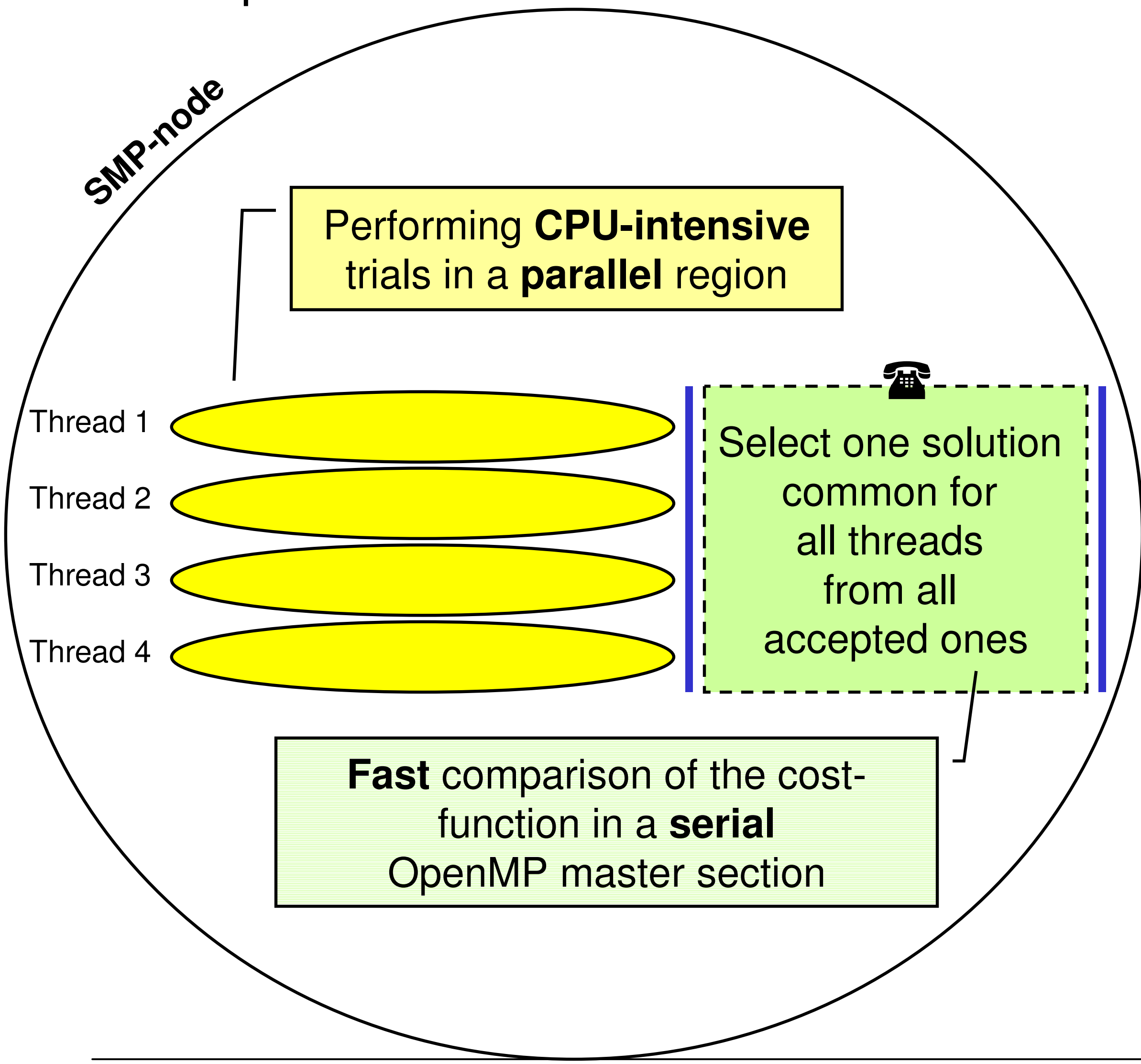
At the end, the best solution is picked up as the final one

The maximal number of engaged nodes is limited by reasonable shortening of the chain length



THE REFERENCE HYBRID COMMUNICATION METHOD

Inner parallelization for communication within nodes




```
for(l=0;l<NoOfTempRed;l++)
{
#pragma omp parallel private(ii)
{
 for(ii=0;ii<EpochLengthPerNode;
 ii=ii+SetOfTrialSize)
 {
#pragma omp for schedule(static)
  for(i=0;i<SetOfTrialSize;i++)
   perform_trial();
#pragma omp barrier
#pragma omp master
  {select one solution common
     for all threads ... }
#pragma omp barrier
 } /*end for (ii=0;...) */
} /*end of parallel region*/
 {reduce the temperature ... }
} /*end for(l=0;...)*/
```

THE METHOD WITH A SINGLE DATA EXCHANGE

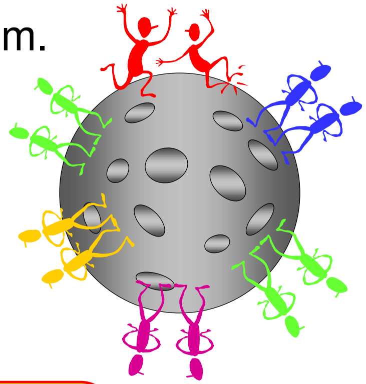
The idea

Incorporating one data exchange after elapsing a percent of the specified time limit (e.g. 50%, 70%)

During the exchange of the data the best solution is selected and mandated for all processes

The idea gives the possibility of:

- Heavy exploration of the search space during the first phase, i.e., a few (but only a few) paths can reach the area of the global minimum.



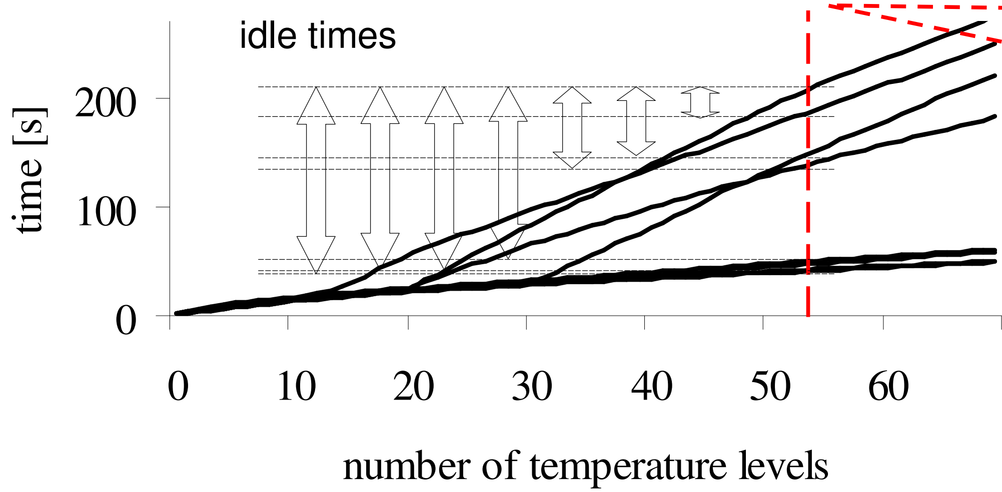
Many small groups of astronauts are looking independently for the deepest crater and hopefully, at least one group (=SMP node) is finding it

- Improvement of the best path during the second phase by all working processes (instead of only a few)

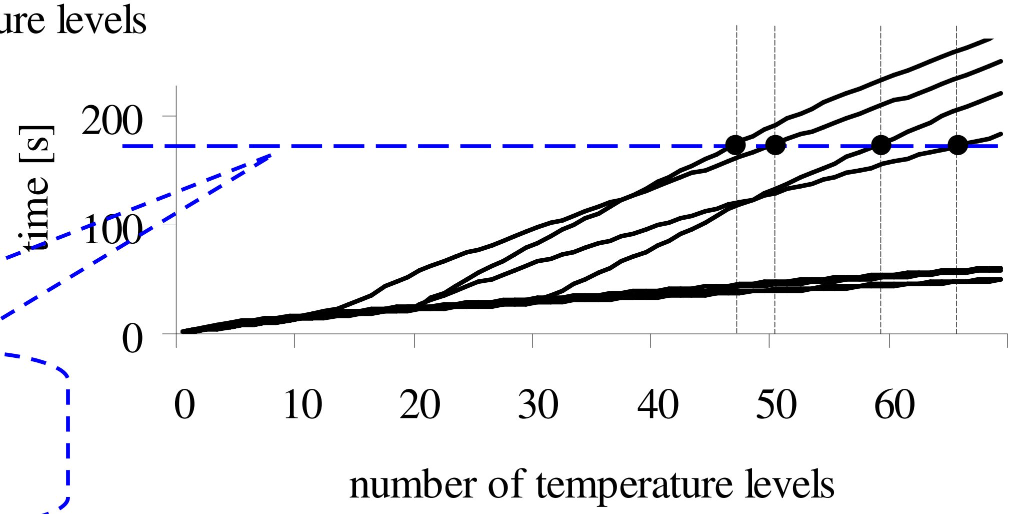
A short time before returning to earth, all groups are concentrated to the deepest crater found up to now.
The last minutes, they **all** try to find the deepest location in that crater!

OUTER LEVEL LOAD BALANCING

The times for generating 8 sub-chains based on an example run



stopping and communicating
after a **fixed**
number of temperature levels



Optimization:

stopping and communicating
after a **fixed**
amount of time

INNER LEVEL LOAD BALANCING

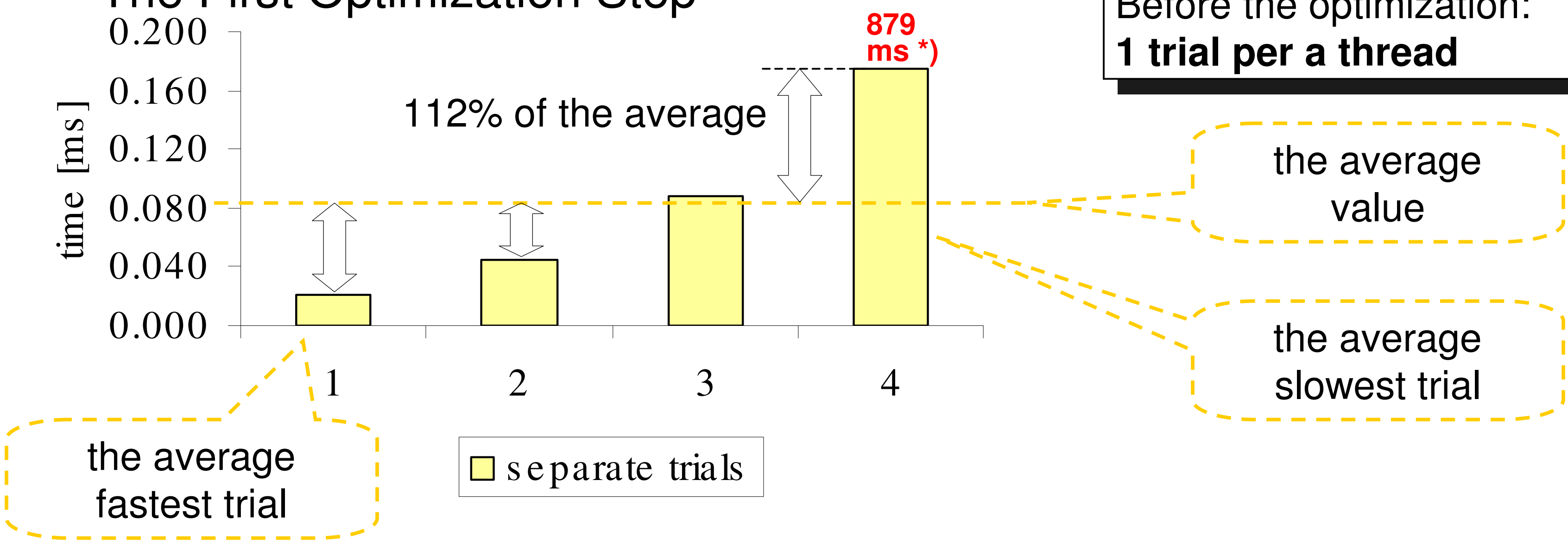
The First Optimization Step

- Each single trial needs extremely different compute time
- Therefore with always 5 trials per thread:
- Better load balancing

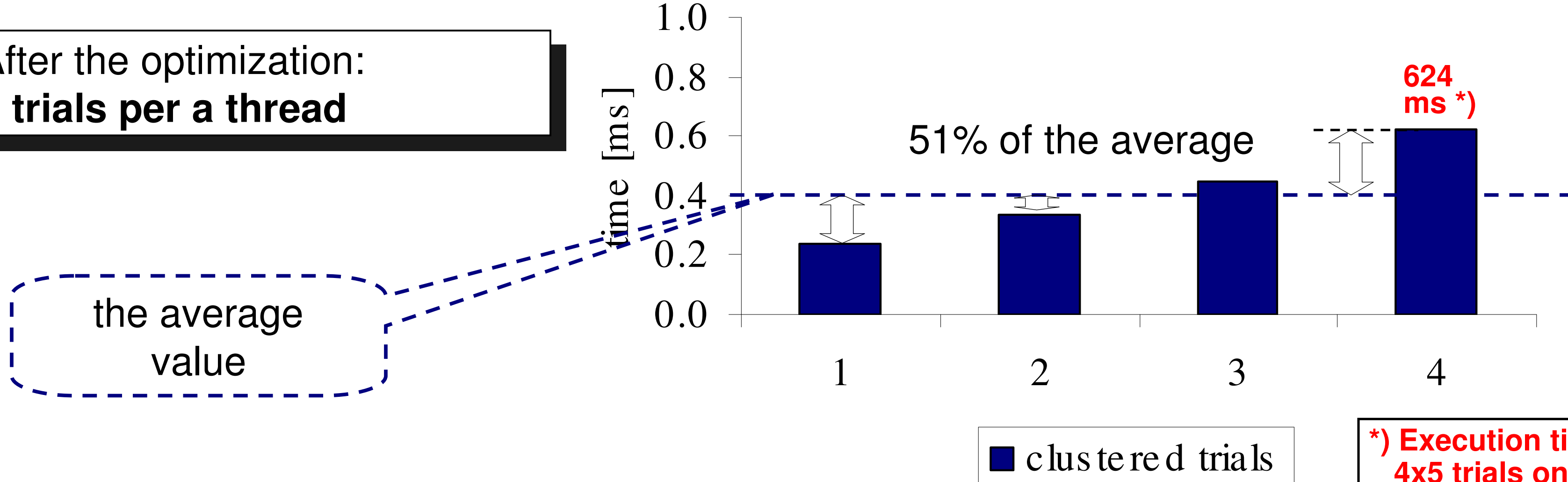
INNER LEVEL LOAD BALANCING

The First Optimization Step

Before the optimization:
1 trial per a thread



After the optimization:
5 trials per a thread



***) Execution time for 4x5 trials on 4 threads**

INNER LEVEL LOAD BALANCING

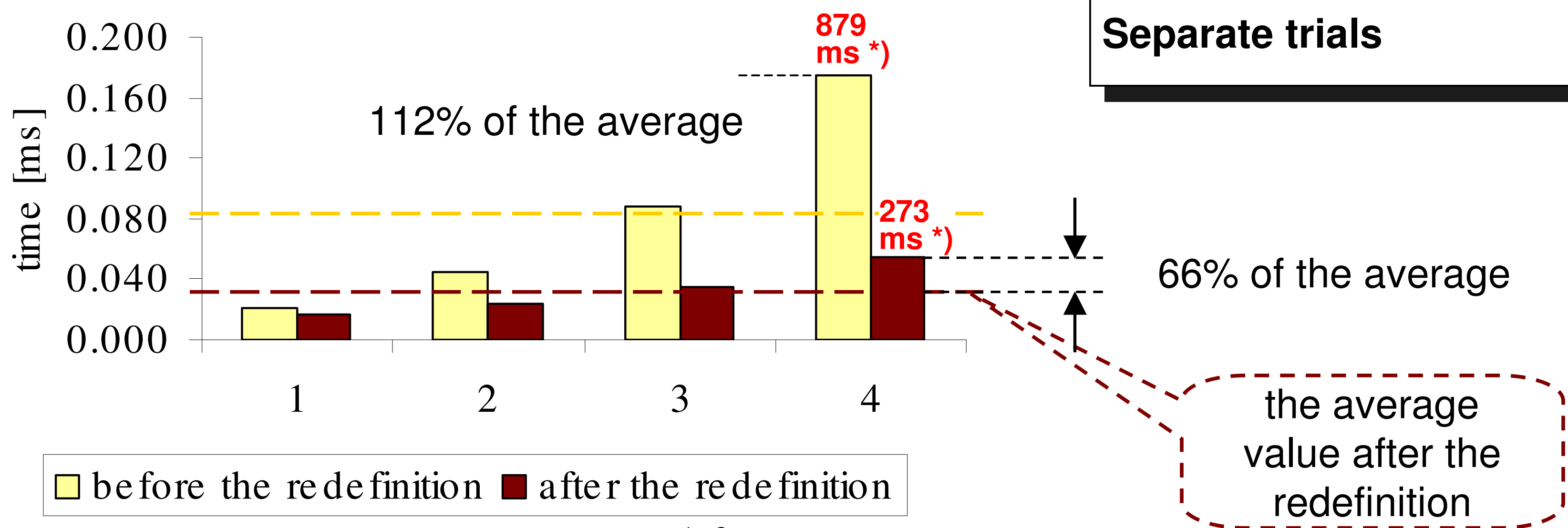
The Second Optimization Step

- Redefinition of a trial:
  - Finding a *new valid solution S‘ in the neighborhood of S*
    - the most time consuming function
  - Allowing a trial to abort this loop without result
    - causes better load balancing
  - Average trial time
    - more dominated by minimal trial size
  - Absolute maximal trial time
    - significantly reduced

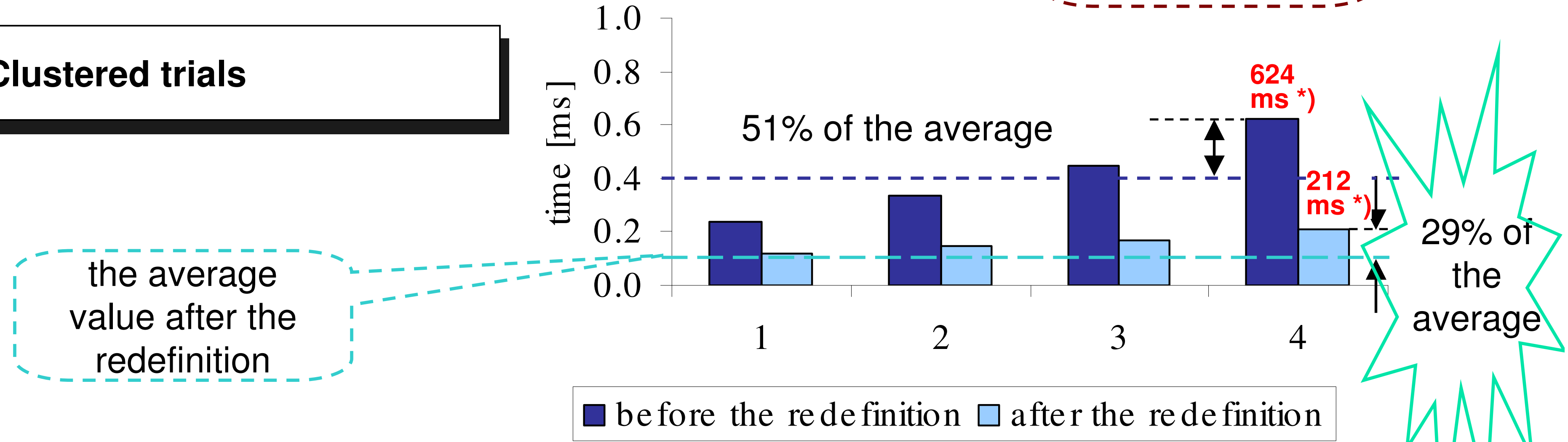
INNER LEVEL LOAD BALANCING

The Second Optimization Step

Separate trials



Clustered trials



*) Execution time for 4x5 trials on 4 threads

INNER LEVEL LOAD BALANCING – The Third Optimization Step

The easiest solution:

```
for(l=0;l<NoOfTempRed;l++)
{
 for(ii=0;ii<EpochLengthPerNode;
 ii=ii+SetOfTrialSize)
 {
#pragma omp parallel for
\schedule(static)

  for(i=0;i<SetOfTrialSize;i++)
   perform_trial();
#pragma omp barrier
/*end of parallel region*/

  {select one solution common
     for all threads ... }

 } /*end for (ii=0;...) */
 {reduce the temperature ... }
} /*end for(l=0;...)*/
```

~40 000

30-40ms *

~20

0.16ms *

Third optimization:

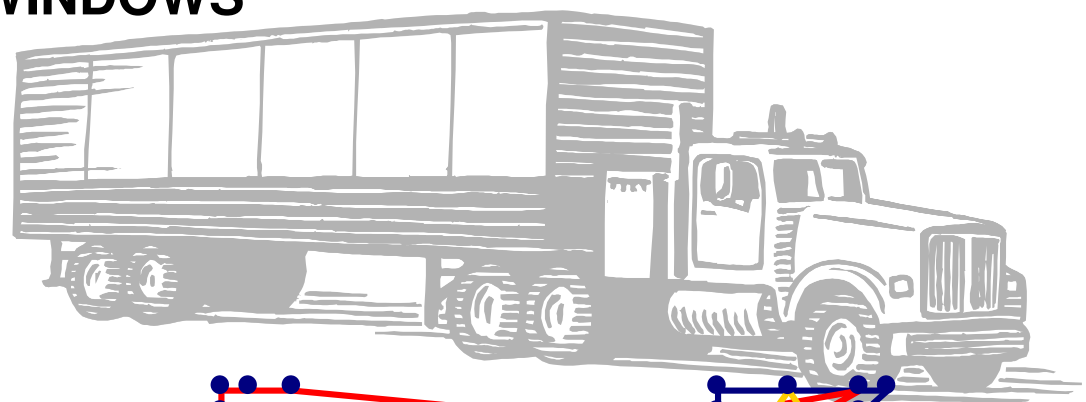
Reducing the fork-join overhead

The optimized solution:

```
for(l=0;l<NoOfTempRed;l++)
{
#pragma omp parallel private(ii)
{
 for(ii=0;ii<EpochLengthPerNode;
 ii=ii+SetOfTrialSize)
 {
#pragma omp for schedule(static)
  for(i=0;i<SetOfTrialSize;i++)
   perform_trial();
#pragma omp barrier
#pragma omp master
  {select one solution common
     for all threads ... }
#pragma omp barrier
 } /*end for (ii=0;...) */
} /*end of parallel region*/
 {reduce the temperature ... }
} /*end for(l=0;...)*/
```


*System: NEC TX7 16x1.5Ghz Intel ItaniumII CPUs, 6MB L3 Cache

VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

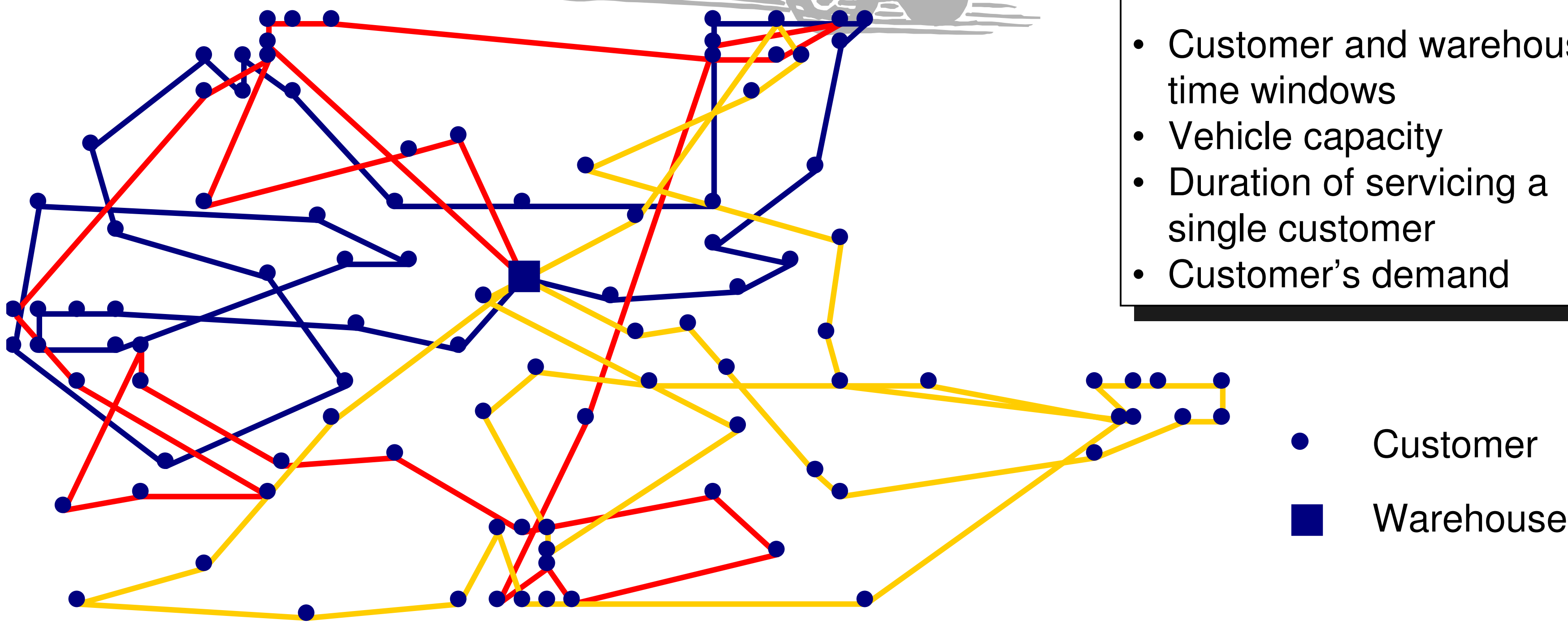


Goal

- Minimizing number of route legs
- Minimizing of the total travel distance

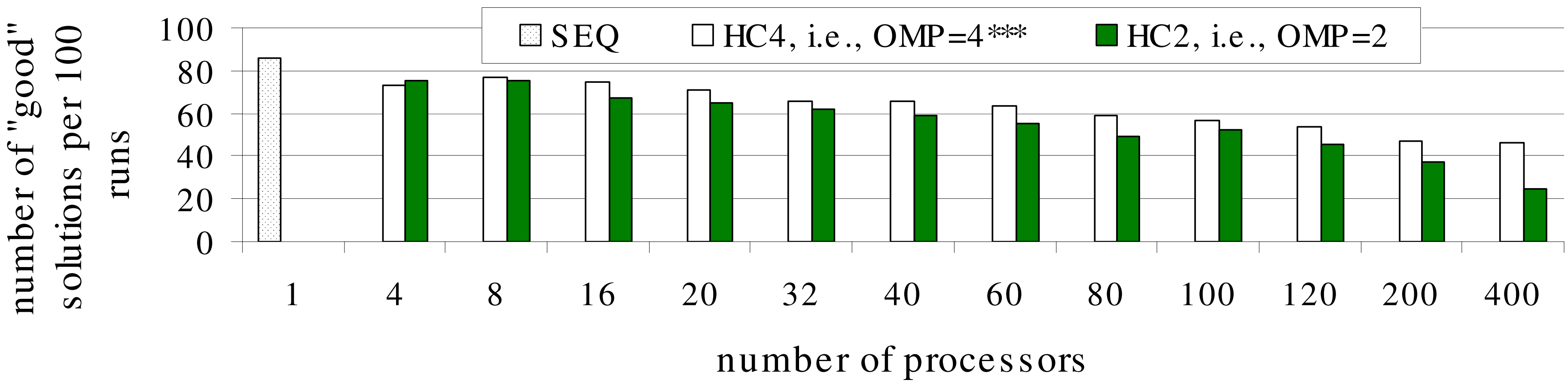
Constraints

- Customer and warehouse time windows
- Vehicle capacity
- Duration of servicing a single customer
- Customer's demand



EXPERIMENTAL RESULTS* – THE FIRST OPTIMIZATION GOAL**

The number of final solutions with the minimal number of route legs (i.e., “good” solutions), generated within 100 runs



Presented values are the averages over the values obtained for 4 data files from Solomon’s benchmark set: R108, R111, RC105, RC108.

Constraints: Execution time x number of CPUs = constant

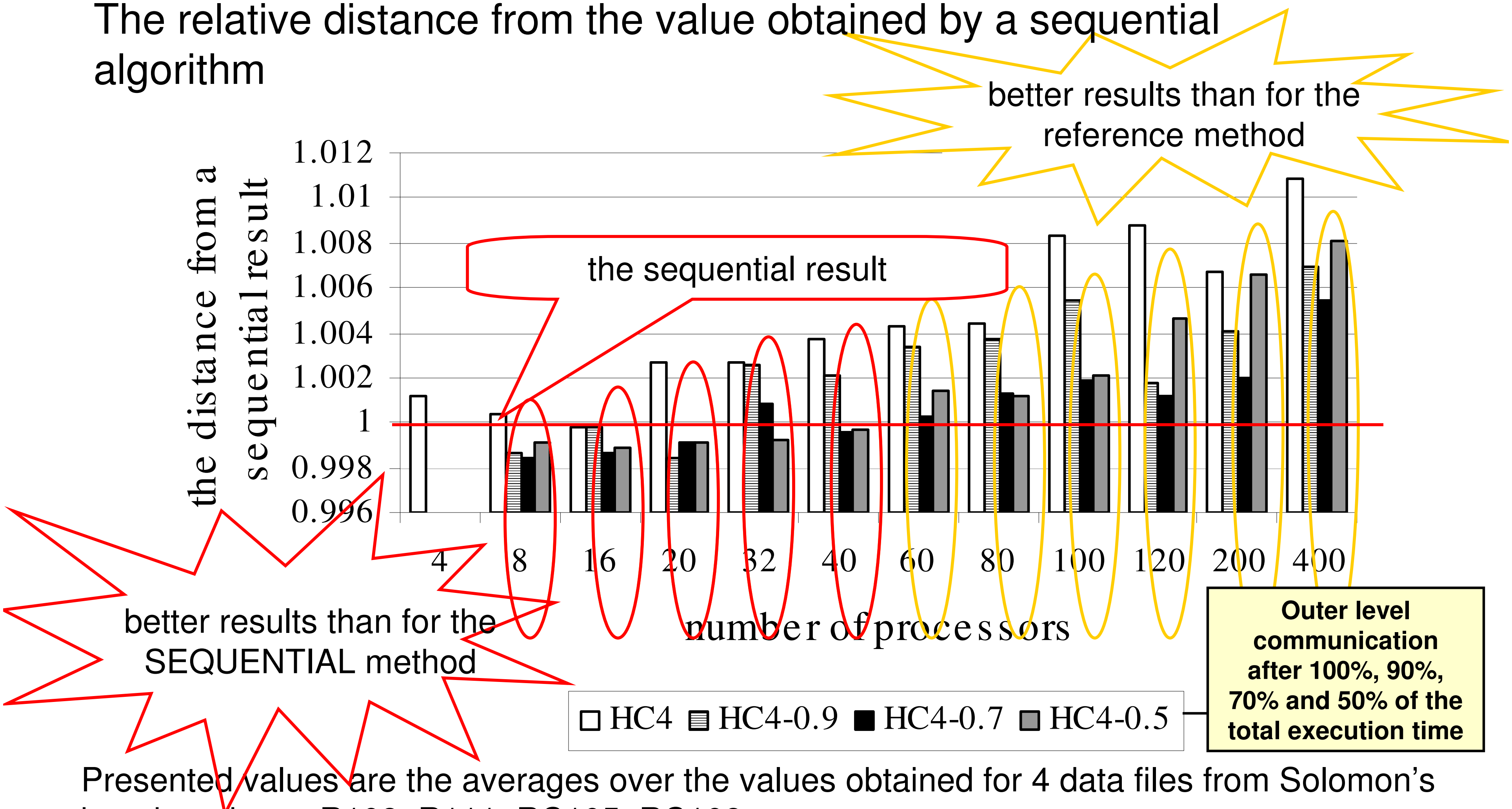
* *Experiments carried out on NEC Xeon EM64T Cluster*

** *The previous work*

*** *Emulated usage of 4 OMP threads, based on results of tests of OMP parallelization carried out on NEC TX-7 system*

EXPERIMENTAL RESULTS – THE SECOND OPTIMIZATION GOAL

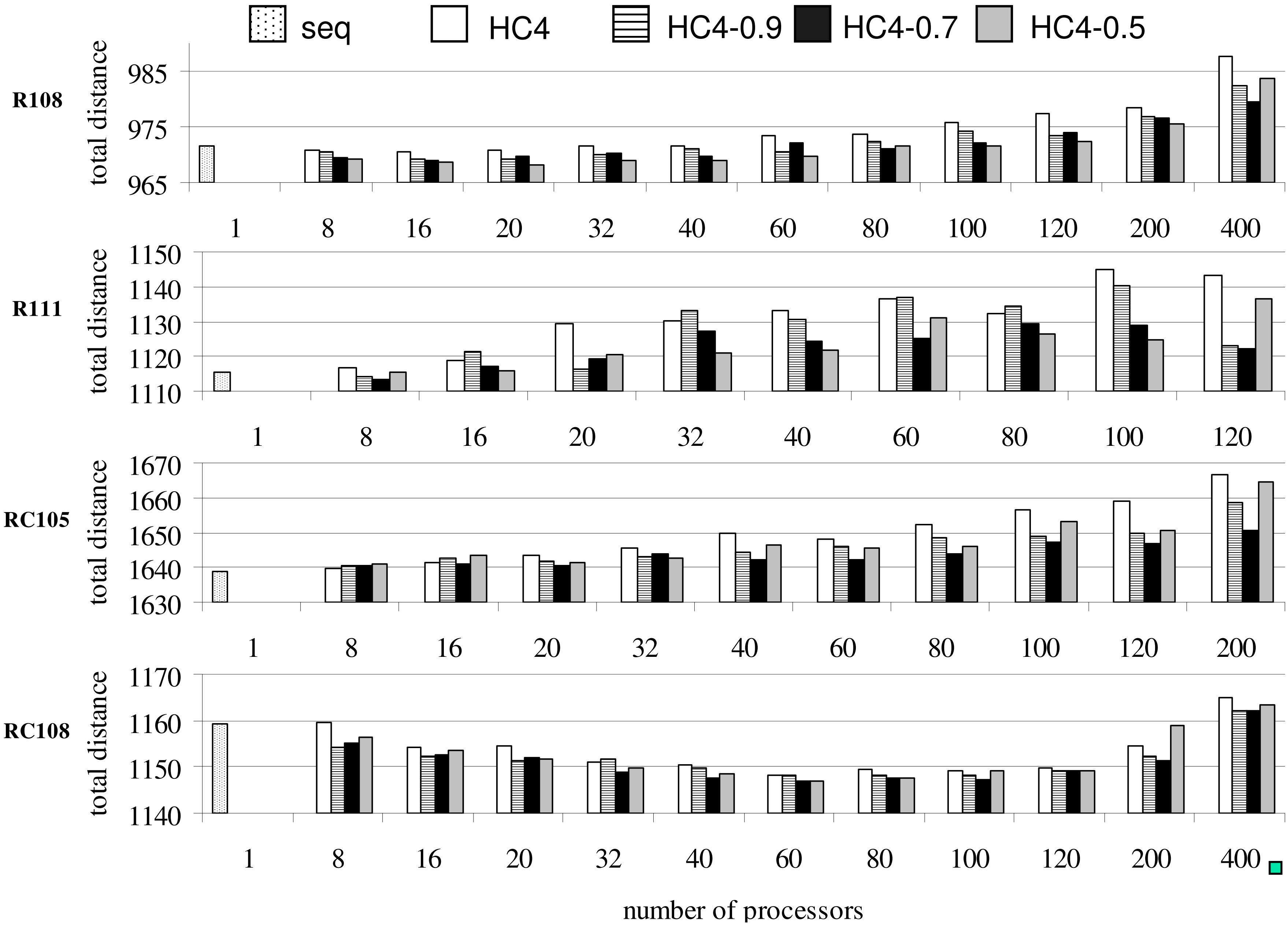
The relative distance from the value obtained by a sequential algorithm



Presented values are the averages over the values obtained for 4 data files from Solomon's benchmark set: R108, R111, RC105, RC108.

Constraints: Execution time x number of CPUs = constant

EXPERIMENTAL RESULTS – THE SECOND OPTIMIZATION GOAL



Acknowledgements

- This project was supported by HPC-Europa.
- The research stay at HLRS Jan. – Feb. 2005 was granted by HPC-Europa.

HPC-Europa

www.hpc-europa.org

Pan-European Research Infrastructure on High Performance Computing at

– HLRS, Stuttgart	– SARA, Amsterdam	– idris, Paris
– epcc, Edinburgh	– CEPBA, Barcelona	– CINECA, Bologna

Research Grants for Computational Science

- Access to HPC
- Scientific Collaboration
- Technical Support
- All **travel and living expense fully reimbursed!**
- 3 – 13 weeks

Poster outside

Applications are welcome at any level, from postgraduate researchers to senior profs.

- **Currently, high acceptance rate**