Using metaheuristics in a parallel computing course

Ángel Luis Calvo
Ana Cortés
Domingo Giménez
Carmela Pozuelo
Miguel Ángel Rodríguez

Funded by Consejería de Educación de la Comunidad de Murcia, Fundación Séneca, project number 02973/PI/05

Course description

- Algorithms and Parallel Computing
  - Fifth year of Computer Science
  - Optional
  - Small classes

  ⇒ high level students, interested in the subject
  - One semester, sequential and parallel parts

  ⇒ two months for parallel computing
  - Proposal of a challenging problem: develop sequential heuristic methods and OpenMP and MPI versions

Syllabus

- Introduction to complexity of problems
- Probabilistic algorithms
- Metaheuristics
- Matricial algorithms
- Models of parallel programming
- Analysis of parallel algorithms
- Parallel algorithms

dedicated to tackling difficult problems

Proposed problem

- A processes to processors mapping in heterogeneous systems

  ⇒ application of metaheuristics and approximated techniques to a parallel computing problem and use of parallelism to improve the solution:

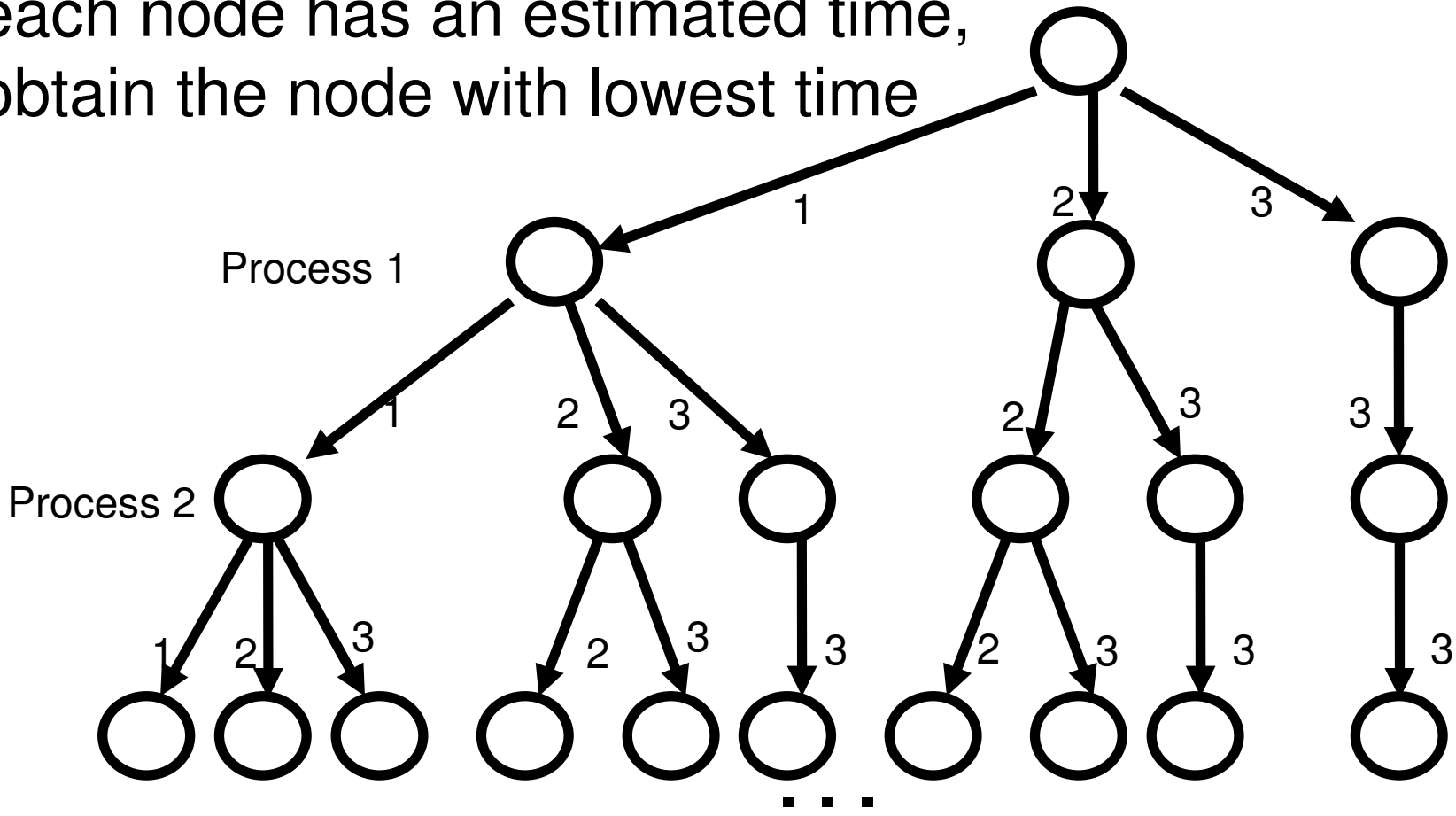
  - Faster execution time
  - Better solution

Methods

- Backtracking: heuristics based pruning, tree traversal guided by heuristics
- Branch and Bound: heuristics based pruning
- Probabilistic algorithms
- Hill climbing
- Tabu search
- Scatter search
- Genetic algorithms
- Ant colony
- Simulated annealing

Mapping problem

Tree of processes to processors mappings, each node has an estimated time, obtain the node with lowest time



Execution time model

- Heterogeneous computation: t_{ci}
- Heterogeneous communication: t_{sij}, t_{wij}
- Execution time:

$$t = t_c\ t_{comp} + t_s\ t_{start} + t_w\ t_{word}$$

$$t_c = \max\{d_i\ t_{ci}\}$$

$$t_s = \max\{d_i \neq 0,\ d_j \neq 0 : t_{sij}\}$$

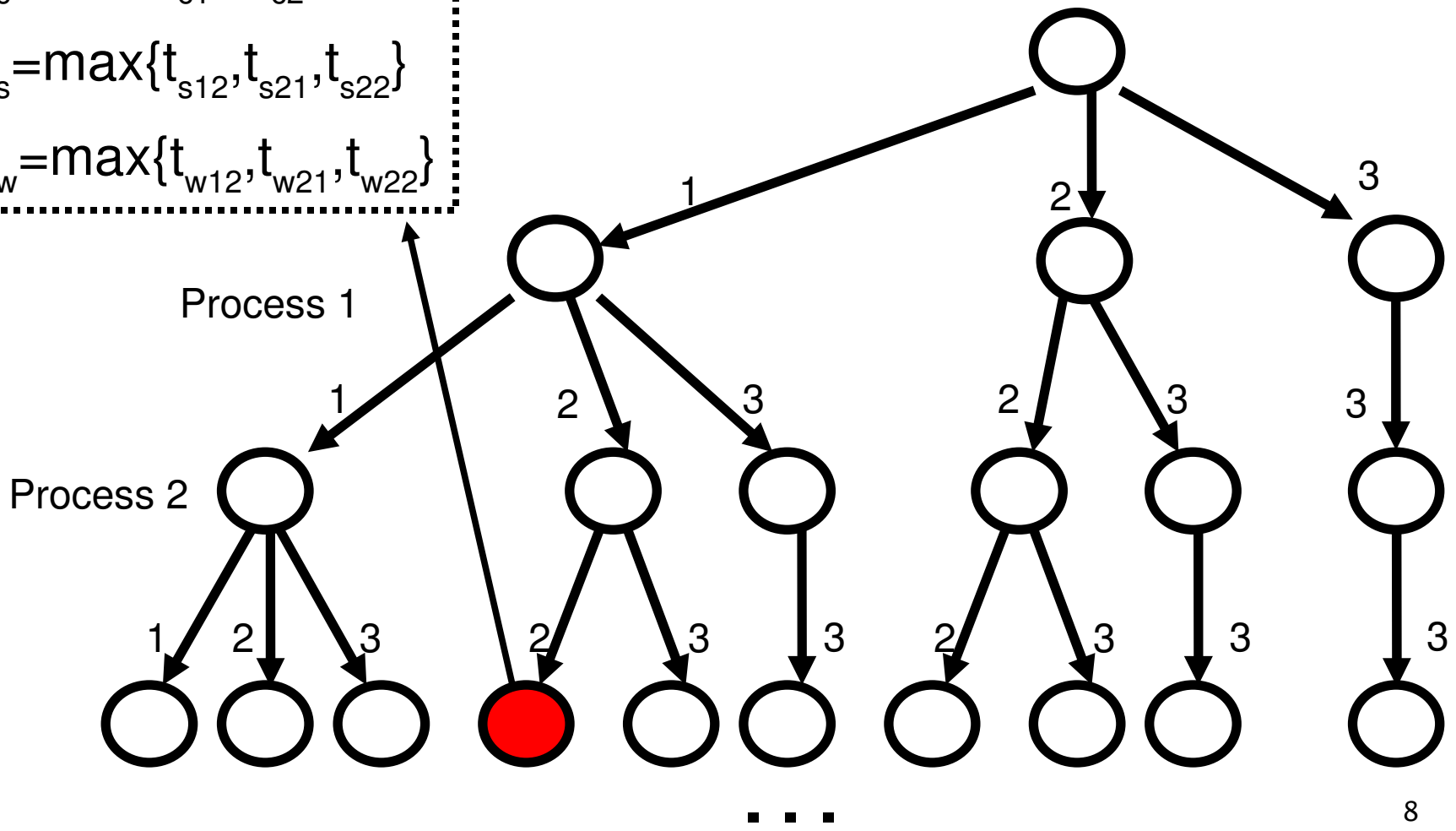
$$t_w = \max\{d_i \neq 0,\ d_j \neq 0 : t_{wij}\}$$

Execution time model

$t_c = \max\{t_{c1}, 2t_{c2}\}$

$t_s = \max\{t_{s12}, t_{s21}, t_{s22}\}$

$t_w = \max\{t_{w12}, t_{w21}, t_{w22}\}$



Backtracking

- Sequential
  - High level scheme: change routines to tune
  - Different pruning techniques: with elimination of nodes which could lead to the optimum
  - From each node greedy estimation of the minimum
- OpenMP:
  - Master generates to a level
  - Slaves do independent backtrackings. Fewer nodes pruned, small reduction in mapping time
- MPI:
  - Like OpenMP. Tasks sent to slaves.

Metaheuristic scheme

```
Initialize(C)
WHILE (NOT
   Convergence(C))
     S = ObtainSubset(C)
     IF |S|>1
          S1= Combine(S)
     ELSE
          S1 = S
     ENDIF
     S2 = Improve(S1)
     C = IncludeSolutions(S2)
ENDWHILE
```

Substitute routines for the particular metaheuristic

Tune routines and parameters to the mapping problem

Genetic algorithm

- Sequential
  - Reduce mapping time with small population and reduced number of iterations to converge
- OpenMP:
  - Parallelize computation of fitness
  - The same mapping
  - Reduction of the execution time for large systems
- MPI:
  - Island scheme
  - No reduction in the execution time
  - Slightly better mapping

Tabu search

- Sequential
  - Satisfactory results when tuning the parameters according to the simulated system (initial assignation to the fastest processors, ...)
- OpenMP:
  - At each step each thread explores one node
  - Slightly better mapping
  - Satisfactory speed-up (superlinear)
- MPI:
  - pC/RS/MPDS technique: each process controls its search; knowledge is not shared; multiple initial solutions; different search strategies
  - Reduction of the execution time if number of iterations is reduced
  - Slightly better mapping

Simulated annealing

- Sequential
  - Small neighbourhood set
  - Satisfactory results with different initial temperatures and cooling functions
- OpenMP:
  - Parallelize computation of fitness
  - The same mapping
  - Reduction in the execution time for large systems
- MPI:
  - Each process carries out a search. The best values are shared periodically
  - Small reduction in the execution time
  - Slightly better mapping

Conclusion, future works

- Work with a challenging problem
- Combine study of heuristic methods and parallel computing
- Enrichment by working on the same problem with different methods
- Other problems:
  - Mapping a tasks graph to a graph representing a hierarchical cluster
  - Mapping a loop of tasks to memory and computational heterogeneous systems