Using Metaheuristics in a Parallel Computing Course

Ángel-Luis Calvo
Ana Cortés
Domingo Giménez
Carmela Pozuelo

University of Murcia, Spain

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

ICCS 2008, Krakow, June 23-26
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
- Tree traversal methods
- 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
- GRASP
Mapping problem I

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:

$$ t = t_{comp} + t_{start} + t_{word} $$

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

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

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

\[ t_c = \max\{t_{c1}, 2t_{c2}\} \]
\[ t_s = \max\{t_{s12}, t_{s21}, t_{s22}\} \]
\[ t_w = \max\{t_{w12}, t_{w21}, t_{w22}\} \]
Mapping problem II

A set of tasks generated by a master processor, with memory restrictions,

nodes eliminated due to memory restrictions

Task 1

Task 2

Task 3
Execution time model

- Heterogeneous computation: $t_{ci}$
- No communications
- Execution time of each node (mapping $d$):
  $$\max_{j=1,...,P} \{ t_{cj} \sum_{l=1,...,T, d_i=j} c_l \}$$
- Optimization problem:
  $$\min_d \max_{j=1,...,P} \{ t_{cj} \sum_{l=1,...,T, d_i=j} c_l \}$$
Course Schedule and Methodology

- Course and problem presentation
  Guide the course with the problem
- Scheduling and mapping problems
- Problems complexity
- Exhaustive tree traversal search
- Parallel systems and parallel programming paradigms
  Sequential methods and parallel programming interleaved
- OpenMP
- MPI
- Probabilistic algorithms
- Metaheuristics
- Matricial algorithms
  A general metaheuristic scheme
  Tutorial: sequential methods
  Presentation: sequential methods
  Practical: sequential methods
- Analysis of parallel algorithms
- Parallel algorithmic schemes
  Tutorial: parallel methods
  Presentation: parallel methods
  Practical: parallel methods
Student work I: 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
Student work I: Genetic algorithm

- **Sequential**
  - Reduced 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
Student work I: Tabu search

- Sequential
  - Satisfactory results when tuning the parameters according to the simulated system (initial assignment 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
Teaching evaluation

- Course organization:
  - Combination of approximation methods and parallel computing
  - Problem guided course

- Questionnaire: Likert scale (1-5), positive and negative items:
  - Suitable combination to study high computational problems
  - Combination means course overloaded
  - Mapping problem helps to clarify metaheuristics
  - The problem in heterogeneous systems means more difficulty
  - The mapping problem clarify ideas of parallel programming
  - Interlacing difficulties the understanding of the themes
  - To guide the course with a problem motivates the study
Questionnaire answers
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
- Problem-guided course

Other problems:
- Mapping a tasks graph to a graph representing a hierarchical cluster
- Generalize the master-slave problem
- Apply the ideas to service search on the web
- ...

...