Parallel Evolutionary Algorithms
(new trends)

University of Málaga, SPAIN

Enrique Alba
**Introduction: The Problem**

- **Objective** of a global optimization problem:

\[
f(\vec{x}) \rightarrow \max: \text{ find a vector } \vec{x}^* \\
\text{such that } \forall \vec{x} \in M : f(\vec{x}) \leq f(\vec{x}^*) := f^*
\]

- **Minimizing** is also possible
- **Vectors** can map to other data structures
Introduction: Real-World Problems

Where can optimization problems be found?
Introduction: Taxonomy of Optimization Algorithms

Optimization Algorithms

- Exact
  - Calculus
    - Direct
      - Newton
    - Indirect
      - B&B
  - Enumerative
    - DP
    - VNS
    - TS
- Ad-hoc Heuristic
  - Trajectory
    - SA
    - VNS
    - TS
  - Population
    - EA
    - ACO
    - PSO

*nature inspired in red*
Introduction: Parallel Algorithms for Optimization

- Basic components of parallel algorithms are:
  - **Nodes** performing separate search
  - **Communication pattern** among the nodes
  - **Policy** of the search (start, end, solution…)

**Ex1:** Parallel ACO
- ACO nodes
- All to all
- Exchange pheromone matrix

**Ex2:** Parallel SA
- SA nodes
- Random target
- Exchange actual best solution

**Ex3:** Parallel EA
- Subpops. of partial solutions
- Static ring
- Exchange random selected sols.
Introduction: Parallel Algorithms for Optimization

Centralized PSO in sequential

1 CPU

Centralized PSO in parallel M/S

1 CPU

Centralized PSO in parallel distribution

CPU1

CPU2

Decentralized PSO in parallel

CPU1

CPU2

CPU4

CPU3

Parallel models: example with PSO
Introduction: Taxonomy

- Three features:
  - Node granularity
  - Central memory
  - Static/Dynamic features
  - …

- Other important issues are:
  - Convergence
  - Synchronization
  - Homogeneity
  - Homogeneity
  - Heterogeneity
  - Async
  - Sync
Research with Parallel Algorithms

Working with parallel algorithms is far from trivial:

1. Must know on concurrency and parallel theory
2. Must know on parallel software languages and tools
3. Must know on hardware realizations for communication
4. Must know on communication protocols and networks
5. Programming is more error prone than in sequential
6. A parallel algorithm is hard to analyze
7. After all the work, maybe you are not gaining in time
8. Many people just don’t want to know on parallel issues
9. ...
...but it is worthwhile!

Problems not solved before become now solvable by using parallel algorithms
Facts in parallel optimization:

1. Model and implementation are different
2. Metrics need a revision
3. Superlinear speedup is a fact
4. Heterogeneity is a must nowadays
5. The experimental setup is important
6. Algorithms are Software
7. Other facts

E. Alba,
Parallel Metaheuristics: A New Class of Algorithms
Fact 1: Model and Implementation are Different

Node in a decentralized EA

1. Generate initial population
2. Evaluate present evaluation
3. While not stop criterion do:
   3.1 Select partners
   3.2 Apply variation operators
   3.3 Communication with neighbors
   3.4 Replace old solutions by the new ones
   3.5 Compute statistics and performances

Index

Introduction

Facts in PO Research

Software and PO

Applications

Case Studies

Summary
Fact 1: Model and Implementation are Different

Decentralized Model

Centralized Model

Index

Introduction

Facts in PO Research

Software and PO

Applications

Case Studies

Summary
Fact 1: Model and Implementation are Different

Decentralized Model

Centralized Model

IMPLEMENTATION-IMPLEMENTATION-IMPLEMENTATION

1 CPU

Cluster of Workstations (COW)
Fact 2: Metrics Need a Revision

- Exact and approximate algorithms could be different algorithms when run in parallel
- Specially, parallel heuristic algorithms can stop at solutions of different quality
- Comparing times against the sequential version could be meaningless if the two algorithms are not exactly the same or the final computed value is different in them
- A clear example is the speedup (efficiency)

A taxonomy is needed

I. Strong Speedup
II. Weak Speedup

A. Speedup with solution-stop
   1. Versus Panmixia
   2. Orthodox

B. Speedup with predefined effort

Index

Introduction

Facts in PO Research

Software and PO

Applications

Case Studies

Summary

Fact 2: Advanced Metrics (Entropy)

Parallel Experiences

**SPH16-32**
8×64 individuals
DPX1\((p_c=1.0)\)
MUT\((p_m=1/l)\)

Index

Introduction

Facts in PO Research

Software and PO

Applications

Case Studies

Summary
Fact 3: Superlinear Speedup is a Fact

\[ S(n_{\text{proc}}) = \frac{T_1}{T_{n_{\text{proc}}}} \]

weak1: against panmixia (1 proc)
weak2: only changing \( n_{\text{proc}} \)

SPH16-32

SSS128
Fact 4: Heterogeneity is a Must Nowadays

Algorithmic components could be heterogeneous

Parallel hardware could be heterogeneous
Fact 4: Heterogeneity is a Must Nowadays

Index

Introduction

Facts in PO Research

Software and PO

Applications

Case Studies

Summary

Parallel Experiences

18 of 50

Fact 4: Heterogeneity is a Must Nowadays
Fact 5: The Experimental Setup is Important

Parallel heuristic/exact methods have often nondeterministic behaviors, so:

1. Multiple independent runs are necessary

2. Statistical hypothesis tests must be used: Student t-test, ANOVA, Wilcoxon, …

3. Average, max and min values need to be reported, but: is that all?

4. Advises: more than 30 ind. runs, give all the parameters in one table, always report on times, give hardware and software used, …
Fact 5: The Experimental Setup is Important: Guidelines

Data sets: \( X_1, X_2, ..., X_n \)
- Normality test (Kolmogorov-Smirnov test)
  - True
    - Normal Variables (Mean comparison, Parametric tests)
      - 2 data
        - Levene test and Student t-test
      - > 2 data
        - Analysis of variance (ANOVA)
          - Post hoc mean comparison tests
            - Equality of Variance (Levene test)
              - True
                - Duncan, Student-Newman-Keuls (SNK), and/or Bonferroni tests
              - False
                - Tamhane tests
            - Non-Normal Variables (Median comparison, Non-Parametric tests)
              - 2 independent data
                - Mann-Witney test
              - 2 dependent data
                - Wilcoxon or Sign tests
              - > 2 dependent data
                - Friedman test
              - > 2 independent data
                - Kruskal-Wallis test
Fact 6: Algorithms Are Software

We should worry about the design of algorithms
  • At the end, they are software pieces (!)
  • Take care of the design and take care of the documentation
  • Efficiency is usually the more important issue

Traditional tools
  • Fortran, C, Java, Lisp, ...
  • Generalization is in conflict with efficiency
  • Apply well-accepted recomms. on GOTO, global vars, ...

Object Orientation
  • Present best issue (long term development and design)
  • Efficiency can be modulated
  • Parallel software tools are developed nowadays
  • Java versus C++

Fact 6: Algorithms Are Software

MALLBA
http://neo.lcc.uma.es
Fact 6: Algorithms Are Software

Complex Data Structures for Complex Problems

IF pos IS NL AND vel IS NL THEN for IS PL

Input layer
- a, b, c, ..., m

Hidden layer
- n, o, p

Output layer
- q, r, s, ..., z

String
- [10, 35, 204, 78, 27, 106]

Interpretation index
- Type I, Type II, Type III, Type IV

Process 1
- P0, P1, P2, ..., Pn

Processor 1
- Processor 0

Processor 2

Parallel Experiences

Index

Introduction

Facts in PO Research

Software and PO

Applications

Case Studies

Summary
• Theory is hard but important: convergence, time complexity, landscape theory, …

• Knowledge exchange between fields helps: developing common frameworks for grid algorithms, exact and heuristic issues, …

• Parallel algorithms are not always better: communication overhead, numerical search could progress slowly, …

• Be always ready for new facts!
There exist multiple software tools to develop parallel applications:

Low level tools: Sockets, …

High Level Comm Libraries: PVM, MPI, …

Language Embedded: Java RMI, …

Middleware: CORBA, MANIFOLD, MR, …

Metacomputing Systems: Globus, Condor, BOINC, …

Related to Internet: .NET, SOAP, XML, …

Platform specific: CUDA, OpenCL, Handle C, …

Others: OpenMP, HPF, …
Our Software

http://neo.lcc.uma.es/software/index/

Welcome to the NEO Software Area

It's our pleasure to welcome you to the NEO Software Area. This Site has been established to give you the ability to easily browse the many software we have developed.

The right menu will direct you to our different softwares and access their home pages. Additionally, most of the sections have an upper menu that allows you to surf the pages of the software you are viewing. Click on the NEO logo in the upper corner if you wish to visit the home page of our server.

If you have any question, or need assistance at any time, please do not hesitate to contact us. You will find details for the responsible staff in some sections.

Enjoy! :-)

Our Software

- Malba
- ssGA
- JGDS
- xxGA
- JCell
- MHTB
- DEME
- JMetal
- More
• **Library for complex optimization problems**
  
  – 3 types of techniques:
    - **Exact**
    - **Heuristic**
    - **Hybrid**
  
  – 3 implementations:
    - **Sequential**
    - **LAN**
    - **WAN**

• **Goals:**
  
  – Wide genericity but low effort of instantiation
  – Simplified utilization (no parallel skills needed)
  – Geographically distributed computing systems

---

Parallel Experiences

Index

Introduction

Facts in PO Research

Software and PO

Applications

Case Studies

Summary

R.O.S.

- **client**
  - **on-line** information
  - problem + solver method in XML format

- **INTERNET**
  - location of antennas, vehicle routing, bioinformatics, etc.

- **server**
  - **solution**

- **search methods:**
  - EA’s, B&B, SA, DP, HC, ...

- **service platforms:**
  - ASP (.NET), J2EE, others

**computational resources**
- [LAN, WAN, Internet]

**sequential or parallel**
- execution of the service

**industrial applications:**
- Location of antennas, vehicle routing, bioinformatics, etc.

**Windows Optimization Service**
- Select Idiom: English, Spanish

**Optimization Service:**
- Start, Configuration, Help
- Open Algorithm Window, New Connection
- Execute, Evaluate Function
- Press Search Server to get the type/server list

**Worker:**
- Running
  - Press Search Server to get the type/server list
Parallel Experiences

Index

Introduction

Facts in PO Research

Software and PO

Applications

Case Studies

Summary

Function Optimization

Generalized Sphere Function

\[
\begin{align*}
\mathcal{f}_{SPH}(\mathbf{x}_{i\mid i=1..n}) &= \sum_{i=1}^{n} x_i^2 \quad x_i \in [-5.12, 5.12] \\
\end{align*}
\]

Rosenbrock’s Function

\[
\begin{align*}
\mathcal{f}_{ROS}(\mathbf{x}_{i\mid i=1..n}) &= \sum_{i=1}^{n-1} 100 \cdot (x_{i+1} - x_i^2) + (x_i - 1)^2 \\
\end{align*}
\]

Domain Dependent Benchmarks

Typical benchmarks for every domain: multiobjective, temporal series, data mining, neural network training, etc.

Rastrigin

\[
\begin{align*}
\mathcal{f}_{RAS}(\mathbf{x}_{i\mid i=1..n}) &= 10 \cdot n + \sum_{i=1}^{n} x_i^2 - 10 \cdot \cos(2 \cdot \pi x_i) \\
\end{align*}
\]

CEC 05/08/10, GECCO 09…
NP-hard Problems (I)

Minimum Tardy Task Problem (MTTP)

Maximum Cut Problem (MaxCut)

Partitioning a weighted graph with "maximum cut"

100 vertices (MaxCut100)

Parallel $\mu$CHC
Satisfiability Problem (MAXSAT)

Goal: Find a feasible assignment for boolean variables

Fitness Function: \[ fitness = \frac{\sum_{i=1}^{m} true(C_i)}{m} \]

Example:

\((\neg A \lor B \lor C) \land (\neg B \lor C \lor D) \land (A \lor \neg D \lor E) \land (C \lor \neg D \lor \neg E)\)

Clauses \((m)\): 4
Variables per Clause \((k)\): 3
Variables \((n)\): 5

Unfeasible Assignment:
\[
\begin{array}{ccc}
A & B & C & D & E \\
1 & 1 & 0 & 1 & 1 \\
\end{array}
\]

Feasible Assignment:
\[
\begin{array}{ccc}
A & B & C & D & E \\
1 & 1 & 1 & 1 & 0 \end{array}
\]

Can theory guide a PEA?
NP-hard Problems (III)

Vehicle Routing Problem (VRP)

GPUs can boost COP !!!
Training Neural Networks

Training
Minimizing the error between expected and actual values. Both, classification and prediction are possible applications.

Structure Optimization

OPTIMUM LINKAGE
Minimizing the number of links

OPTIMUM LAYER DESIGN
Optimizing the number of layers, neurons per layer and transfer functions in neurons

Pattern Filtering
Find the minimum number of patterns
Telecommunications

Index

Introduction
Facts in PO Research
Software and PO
Applications
Case Studies
Summary

Parallel Experiences

Radio Network Design

GSM Frequency Assignment

Sensor Network Layout

Location Area in 2G/3G

Covered Area: 85.8% #Transmitters: 56
Covering Codes: 0 1 2 3 +
Parallel Experiences

Index

Introduction
Facts in PO Research
Software and PO

Applications

Case Studies
Summary

MANETs
Parallel Experiences

VANETs

http://diricom.lcc.uma.es

Information Dissemination

Index

Introduction
Facts in PO Research
Software and PO
Applications
Case Studies
Summary
Parallel Experiences

Index

Introduction
Facts in PO Research
Software and PO
Applications
Case Studies
Summary

Miscellaneous

- Designing VLSI Circuits
- Data Based (Data Mining, Query Optimization)
- Dynamic Optimization Problems (DOP)
- Tasks Scheduling in Operating Systems
- Genomics (Fragment assembly, protein structure)

Games:
- Master Mind,
- Unreal Tournament, ...
### Globus and Multiobjective: First Case Study

**Optimize**

\[ f_m(\vec{x}) \]

**Subject to**

\[ g_j(\vec{x}) \geq 0, \quad j = 1, 2, \ldots, j \]

\[ h_k(\vec{x}) = 0, \quad k = 1, 2, \ldots, k \]

<table>
<thead>
<tr>
<th>a</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>

| a | 3 | 7 | 4 | 8 |
|---|---|---|---|
| b | 2 | 1 | 2 | 5 |

**A is better than B**

<table>
<thead>
<tr>
<th>a</th>
<th>1</th>
<th>9</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>3</td>
<td>6</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>

**B is better than A**

<table>
<thead>
<tr>
<th>a</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>

**None is better**
Globus and Multiobjective: First Case Study

Base pseudocode of the enumerative algorithm

1. F[M] = \{F1, F2, ..., FM\} // Objective functions
2. R[C] = \{R1, R2, ..., RC\} // Constraints
3. x[N] = \{x1, x2, ..., xN\} // Decision variables
4. f[M] = \{f1, f2, ..., fM\} // Function values
5. P = \{\} // Set of non-dominated solutions
6.
7. Fix the discretization degree G of the decision variables
8. For each vector x[i]
9. If x[i] satisfies the constraints R[C]
10. f[j] = evaluation of x[i] by F[M]
11. Compare f[j] with members of P for dominance
12. If f[j] is a non-dominated solution
13. Add f[j] to P
14. Remove the solutions dominated by f[j] from P
Globus and Multiobjective: First Case Study

Kursawe
Min $F = \left( f_1(\bar{x}), f_2(\bar{x}) \right)$

\[
\begin{align*}
f_1(\bar{x}) &= \sum_{i=1}^{n-1} \left( -10e^{-0.2\sqrt{x_i^2 + x_{i+1}^2}} \right) \\
f_2(\bar{x}) &= \sum_{i=1}^{n} \left( |x_i|^a + 5 \sin(x_i)^b \right)
\end{align*}
\]

$-5 \leq x_i \leq 5$
$i = 1, 2, 3$
$a = 0.8$
$b = 3$

Golinski
Min $F = \left( f_1(\bar{x}), f_2(\bar{x}) \right)$

\[
\begin{align*}
f_1(\bar{x}) &= 0.7854x_1x_2^2 \left( 10x_2^2/3 + 14.933x_3 - 43.0934 \right) \\
&\quad - 1.508x_1(x_2^3 + x_2) + 7.477(3x_2^3 + x_2^2) + 0.7854(x_4x_6^2 + x_5x_7^2) \\
f_2(\bar{x}) &= \sqrt{\frac{(745.0x_4/x_3)^2 + 1.69 \times 10^7}{0.1x_6^3}}
\end{align*}
\]

\[
\begin{align*}
\frac{1.0}{x_1x_2^2} - \frac{1.0}{27.0} &\leq 0; \quad \frac{1.0}{x_1x_2^2} + \frac{1.0}{27.0} \leq 0 \\
\frac{x_3^4}{x_2x_3^4} - \frac{1.0}{1.93} &\leq 0; \quad \frac{x_3^4}{x_2x_3^4} + \frac{1.0}{1.93} \leq 0 \\
x_2x_3 - 40 &\leq 0; \quad x_1x_2 - 12 \leq 0 \\
5 - x_1/x_2 &\leq 0; \quad 1.9 - x_4 + 1.5x_6 \leq 0 \\
1.9 - x_4 + 1.1x_7 &\leq 0; \quad f_2(x) \leq 1300 \\
\sqrt{\frac{(745.0x_3/x_2x_3)^2 + 1.575 \times 10^6}{0.1x_4^3}} &\leq 1100 \\
2.6 \leq x_1 \leq 3.6; \quad 0.7 \leq x_2 \leq 0.8 \\
17.0 \leq x_3 \leq 28.0; \quad 7.3 \leq x_4 \leq 8.3 \\
7.3 \leq x_5 \leq 8.3; \quad 2.9 \leq x_6 \leq 3.9 \\
5.0 \leq x_7 \leq 5.5
\end{align*}
\]
Globus and Multiobjective: First Case Study

GLOBUS LAYERED ARCHITECTURE

Applications

High-level Services and Tools
- DUROC
- MPI
- Condor-G
- HPC++
- Nimrod/G
- globusrun

Core Services
- Nexus
- GRAM
- I/O
- MDS
- GSI
- GSI-FTP
- HBM
- GASS

Local Services
- Condor
- MPI
- LSF
- PBS
- NQE
- Linux
- NT
- Solaris
- TCP
- UDP
- DiffServ

Parallel Experiences

Introduction
Facts in PO Research
Software and PO
Applications
Case Studies
Summary
Globus and Multiobjective: First Case Study

- CPU Usage (%): 0, 20, 40, 60, 80, 100
- Time
- Network Traffic (KBytes): 0, 500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000
- Time

Index
- Introduction
- Facts in PO Research
- Software and PO
- Applications
- Case Studies
- Summary
# Parallel Experiences

## Globus and Multiobjective: First Case Study

### KURSAWE

<table>
<thead>
<tr>
<th>% time</th>
<th>Subtask 1</th>
<th>Subtask 2</th>
<th>Subtask 3</th>
<th>Subtask 4</th>
<th>Subtask 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraints</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>Dominance</td>
<td>94.9%</td>
<td>96.3%</td>
<td>97.2%</td>
<td>95.3%</td>
<td>93.1%</td>
</tr>
<tr>
<td>Fitness</td>
<td>5.1%</td>
<td>3.6%</td>
<td>2.8%</td>
<td>4.7%</td>
<td>3.9%</td>
</tr>
</tbody>
</table>

### GOLINSKI

<table>
<thead>
<tr>
<th>% time</th>
<th>Subtask 1</th>
<th>Subtask 2</th>
<th>Subtask 3</th>
<th>Subtask 4</th>
<th>Subtask 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraints</td>
<td>66.2%</td>
<td>68.9%</td>
<td>68.9%</td>
<td>68.6%</td>
<td>44.1%</td>
</tr>
<tr>
<td>Dominance</td>
<td>33.8%</td>
<td>31.03%</td>
<td>31.1%</td>
<td>31.4%</td>
<td>55.1%</td>
</tr>
<tr>
<td>Fitness</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>0.8%</td>
</tr>
</tbody>
</table>
Globus and Multiobjective: First Case Study

![Graph showing network traffic (MB/sec) vs. number of tasks.](image)
New Hardware: 2nd. Case Study

GPUs and Multicore systems are revitalizing the research in parallel algorithms

Grid and cloud computing

- GPUs
- Multicore systems
- Grid and cloud computing
- FPGA
- GPU
- multicore
New Hardware: 2nd. Case Study

**Genetic Algorithm + Systolic Computing = Systolic Genetic Search**

- **Systolic Genetic Search (SGS)**
  - Solutions sync. pumped in a grid
  - When two solutions meet in a cell, adapted evolutionary operators
  - New solutions continue moving along the grid

- **Each cell**
  - Applies crossover and mutation on different pre-computed positions
  - Use elitism (there is no selection)
Summary of Challenges

1. Parallel algorithms are not fast versions of sequential algorithms: they are new algorithms
2. Create standard benchmarking for parallel optimization
3. Develop a uniform experimental setup methodology (metrics and stats)
4. Connect the algorithm field to the application and software fields
5. Tackle new issues in grid/cloud/ambient computing
6. New research fields: GPGPU, Multicores, programming
7. Create a body of knowledge in parallel algorithmics
End of Presentation

Málaga

http://neo.lcc.uma.es