Escuela Técnica Superior de Ingeniería Informática
UNIVERSIDAD DE MÁLAGA

EVOLVE, May 2011

Metaheuristic Applications to Telecoms, Bioinf, Software, and other Domains

http://neo.lcc.uma.es

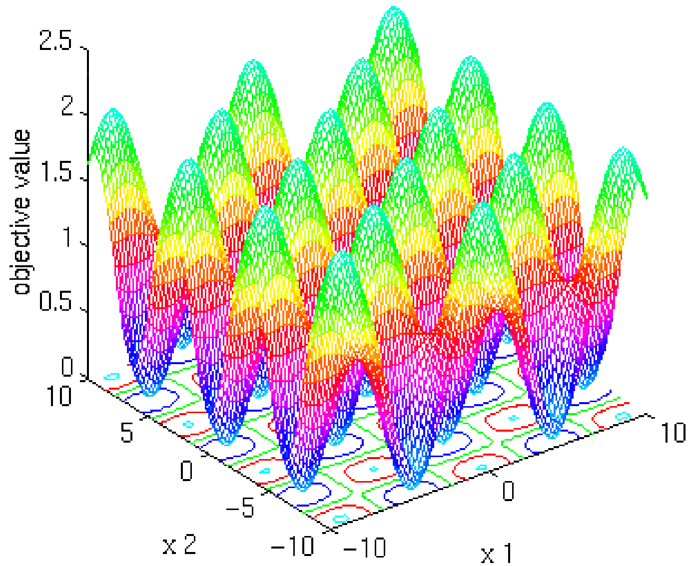
University of Málaga, Spain

Index

Introduction (I)

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

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

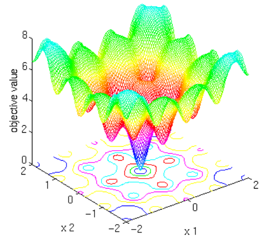


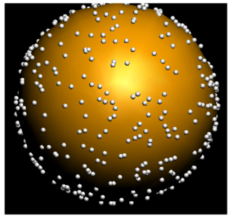
- **Minimizing is also possible**
- **Vectors can map to other data structures**

Index

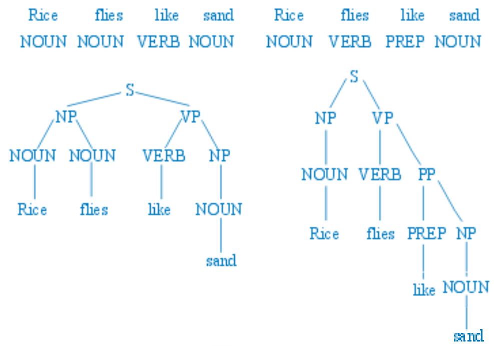
Introduction (II)

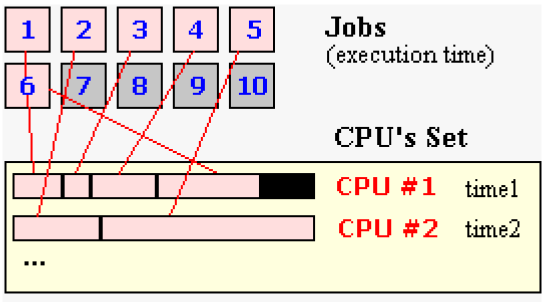
Where can optimization problems be found?

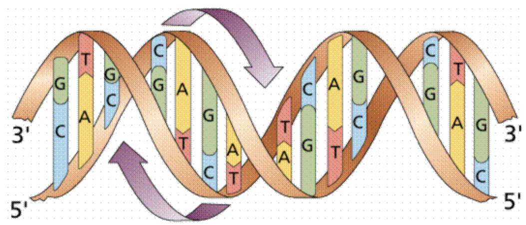


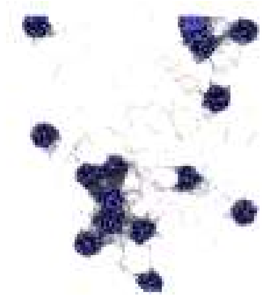


Distribute "n" electrons on a sphere









Index

- New Models
- Parallel MetaH
- Hybrid MetaH
- Multiobjective
- Telecoms.
- Bioinformatics
- Software
- Others

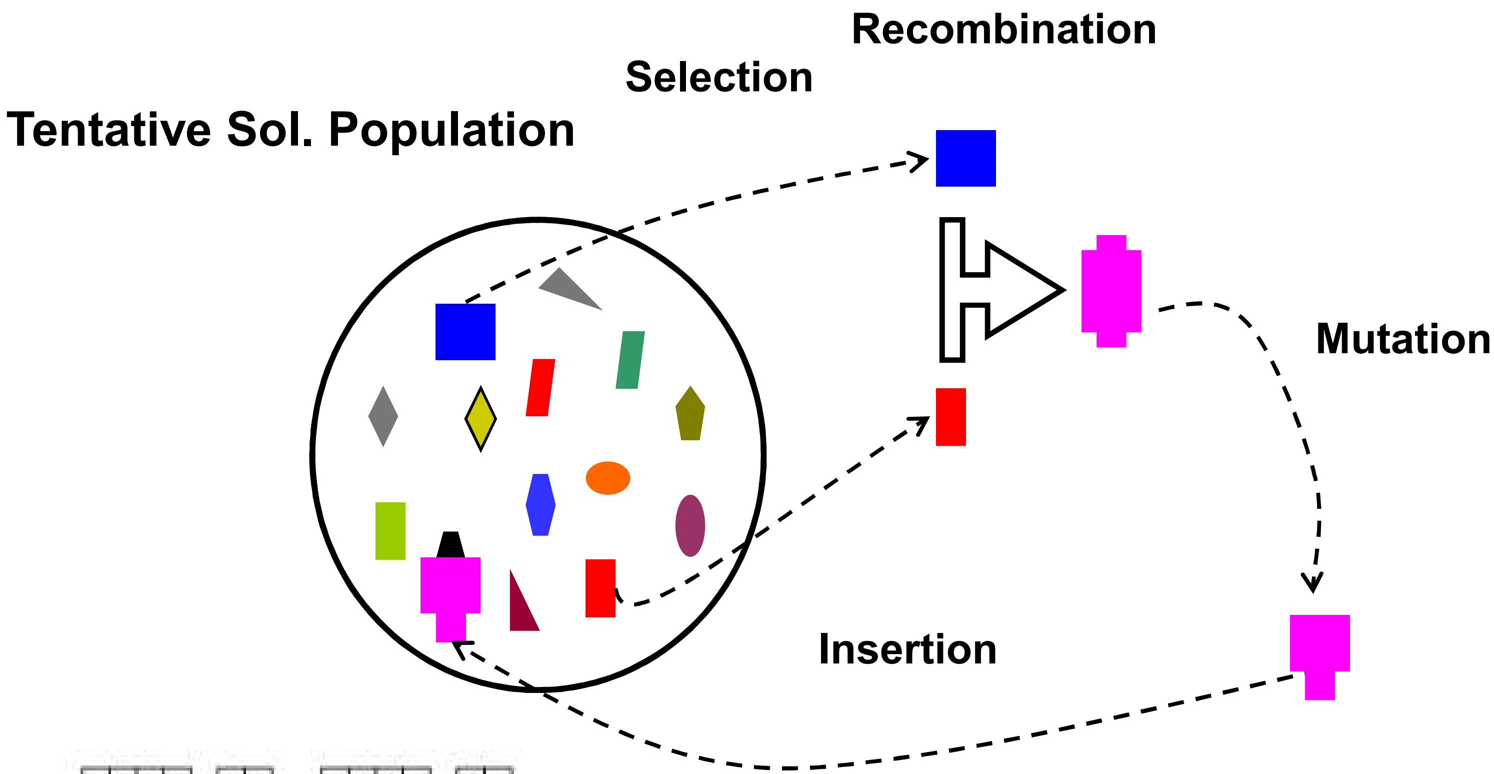
Introduction (III)

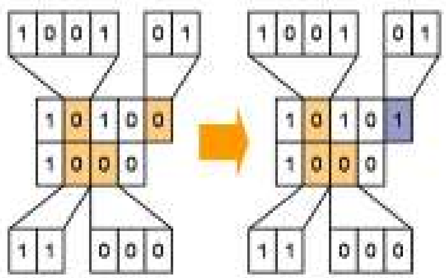
Optimization Algorithms

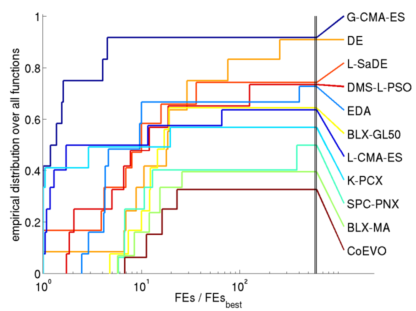
- **Exact**
  - **Calculus**
    - **Direct**
      - Newton
      - Greedy
    - **Indirect**
  - **Enumerative**
    - DP
    - B&B
- **Ad-hoc Heuristic**
- **Metaheuristic**
  - **Trajectory**
    - SA
    - VNS
    - TS
  - **Population**
    - EA
    - ACO
    - PSO

• nature inspired in red

Introduction (IV)





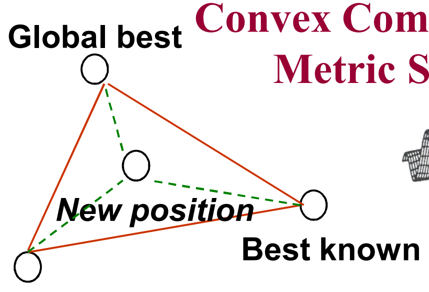


Introduction (V)

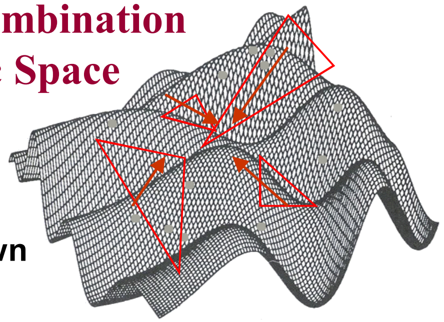
Evolutionary Algorithm

$t := 0$;
`initialize` $[P(t)]$;
`evaluate` $[P(t)]$;
while `not end condition` **do**
    $P'(t) :=$ `variation` $[P(t)]$;
    `evaluate` $[P'(t)]$;
    $P(t+1) :=$ `selection` $[P'(t) \cup Q]$;
    $t := t + 1$;
end while

Introduction (VI)

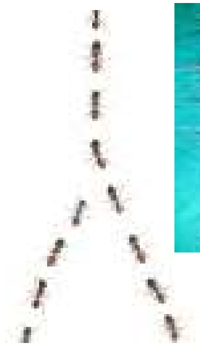


Convex Combination
Metric Space

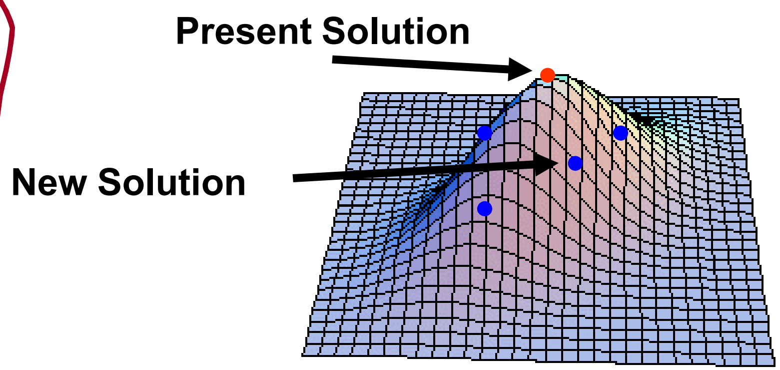


```
procedure ACOMetaheuristic
    ScheduleActivities
        ConstructAntsSolutions
        UpdatePheromones
        DaemonActions // optional
    end ScheduleActivities
end procedure
```

Inspiration



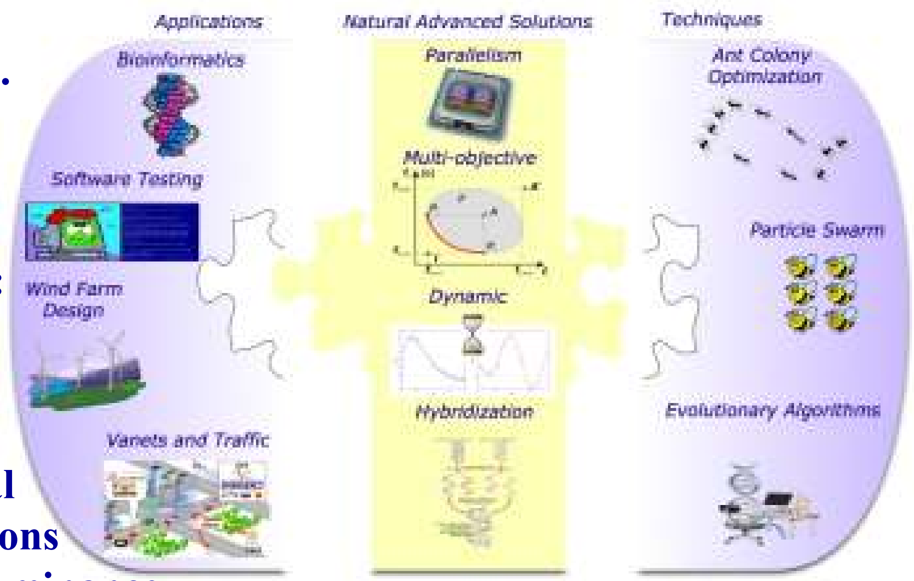
(0,2; -1,4; 3,5)	→ Solution Vector
(1,0; 10,3;	→ Standard Deviation
(1,7; 0,3; 2,1)	→ Search Angles



Introduction (VII)
NAS: Natural Advanced Solutions

Four main ways of making an algorithm more efficient and accurate:

- **Parallel:**
  Clusters, Grid computing, multicore, FPGAs, GPUs…
- **Hybrid:**
  Combining algorithms, operators, representations: problem knowledge
- **Multiobjective:**
  Modelling explicitly several conflicting objective functions with Pareto's concept of dominance



- **Dynamic:**
  Solve a problem that changes in time and adapt previous solutions to the new scenarios

Parallel Metaheuristics

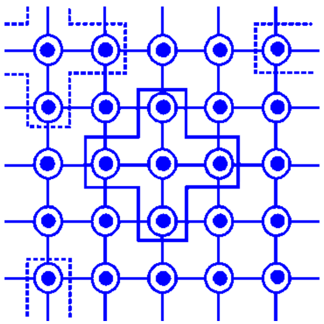
Paralellism and Metaheuristics:

The increasing availability of new kinds of CPUs and the parallel nature of metaheuristics have allowed the fast development of parallel metaheuristics

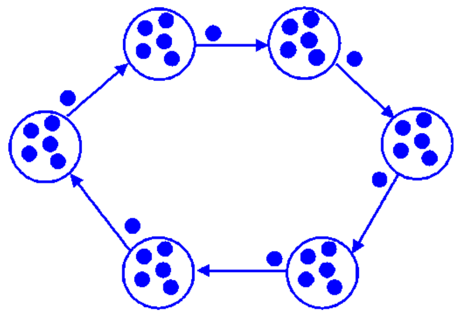
Advantages:

- Allow to tackle more complex problems and/or larger instances
- Allow to reduce the execution time
- Allow to improve the quality of the found solutions

GPU



Examples

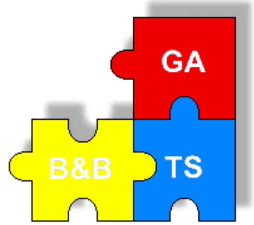


multicore

E. Alba (ed.), Parallel Metaheuristics: A New Class of Algorithms, *Wiley & Sons*, 2005

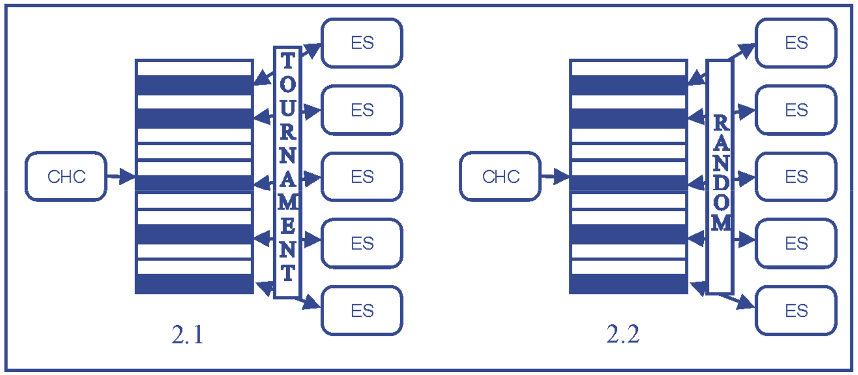
Hybrid Algorithms

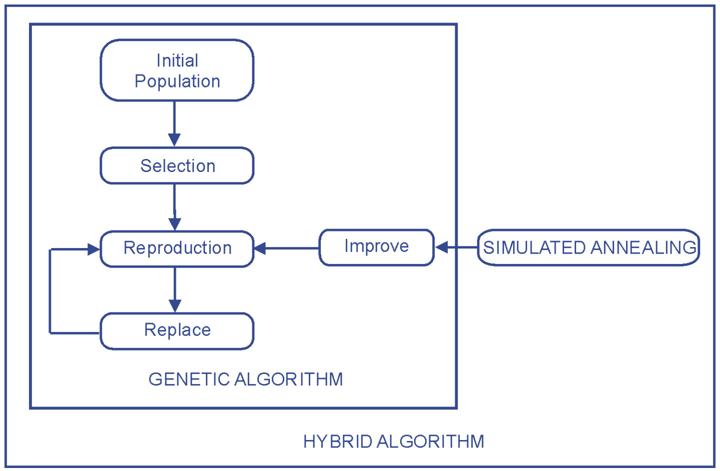
Hybridization is the inclusion of problem-dependant information in the algorithm



Types:

- **Strong**
- **Weak**





Examples

Index

Multiobjetive Optimization

Most real word optimization problems require to optimize more than one single function

- **Multiobjective Optimization Problems (MOPs)**

Multobjective optimization searches for a set of solutions

- **Pareto Optimal Set**
- **Their representation in the objective space is known as Pareto front**

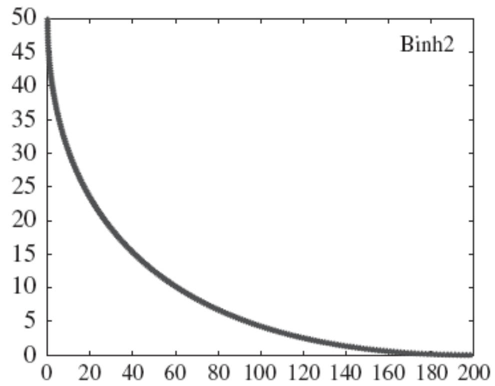
Metaheuristics provide a subset of the Pareto optimal set. Two goals

- **Convergence to the true Pareto front**
- **Diversity of the solutions along the true Pareto front**

$$
\begin{aligned}
\text{Min } F &= (f_1(\vec{x}), f_2(\vec{x})) \\
f_1(\vec{x}) &= 4x_1^2 + 4x_2 \\
f_2(\vec{x}) &= (x_1-5)^2 + (x_2-5)^2
\end{aligned}
$$

Subject to:

$$
\begin{aligned}
g_1(\vec{x}) &= (x_1-5)^2 + x_2^2 - 25 \le 0 \\
g_2(\vec{x}) &= -(x_1-8)^2 - (x_2+3)^2 + 7.7 \le 0 \\
0 &\le x_1 \le 5 \\
0 &\ge x_2 \le 3
\end{aligned}
$$



Multiobjetive Optimization: Open Issues

- Fitness assignment
    - Multiobjective metaheuristics assign a unique value to the solutions used as "fitness" to compare solutions
    - E.g.: Ranking in NSGA-II or strength in SPEA2

- Maintaining diversity
    - Additional information is needed to know the density of solutions around a given one
    - E.g.: Hypercube in PAES, crowding distance in NSGA-II

- Elitism
    - The general approach uses an auxiliary population, sometimes called *archive*

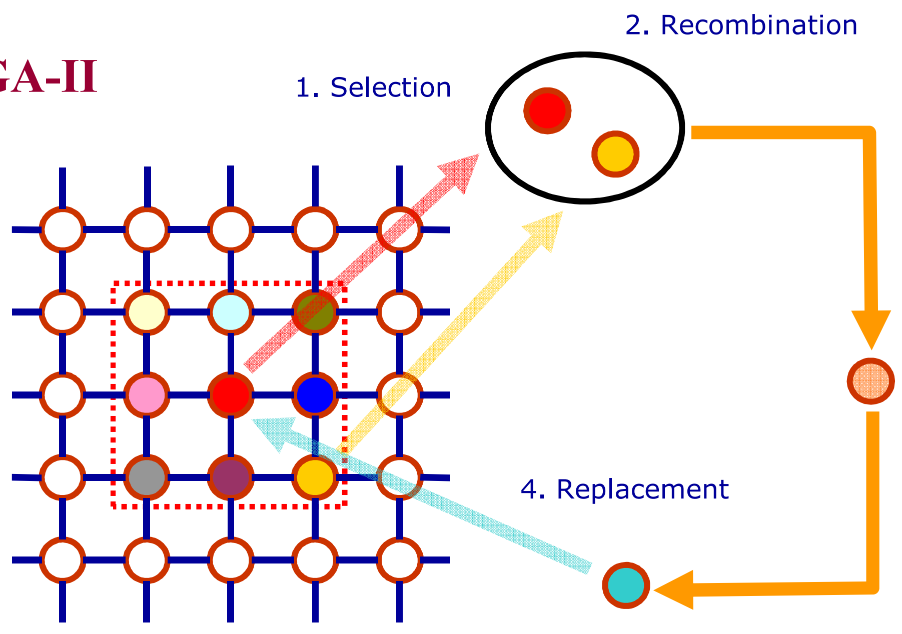
- Quality indicators
    - Metrics are needed to measure convergence and/or diversity
        - ➢ Hypervolume (convergence and diversity)
        - ➢ Generational Distance (convergence)
        - ➢ Spread (diversity)

MOCell

MOCell
- Multiobjective cellular genetic algorithm

Main features
- Use of an external archive
  - 2-dimensions toroidal grid
  - Archive feedback

Comparison against NSGA-II and SPEA2
- Competitive results in terms of convergence and hypervolume
- Best results concerning spread



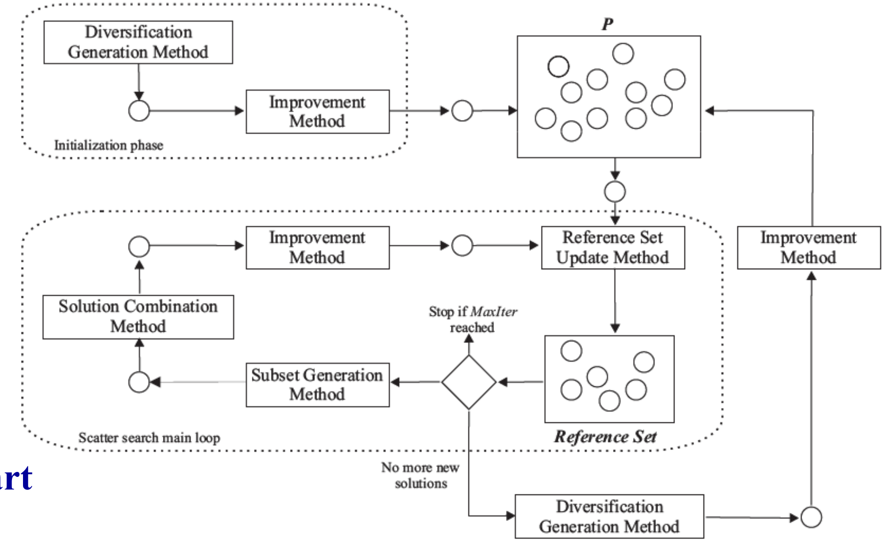
AbYSS

AbYSS

- Archive based hYbrid Scatter Search

Basic idea

- Redefining the scatter search template to adapt it to multiobjective optimization



Main features

- External Archive to maintain good solutions
- Individuals of the external archive are moved to initial set in the re-start loop

Comparison against NSGA-II and SPEA2

- Competitive results in terms of convergence and hypervolume
- Best results concerning spread

Multiobjective Optimization and Grid Computing

Motivation:

- There are many computers in the labs of the Computer Science Departament of the University of Málaga
  - Currently we directly control up to 400 processors
- Question: How can we use them together to solve multiobjective optimization problems?
- Using known message passing libraries (sockets, PVM, MPI) is not a solution
  - Machines are idle in the nights and during the weekends (and in holydays)
  - Variable availability during the day
- OUR APPROACH: using grid technologies

Grid computing systems used:

- Condor
- Globus
- Others: ProActive, BOINC

Issues of interest:

- Easiness of installation and administration
- Parallel programming models offered
- Programming languages available
- Use of idle CPU cycles (opportunistic computing)
- Parallel performance

Index

Optimal Broadcasting in MANETs

- MANETs

- **Stations usually are laptops, handholds, PDAs, or mobile phones**
- **Mobility of stations → dynamic topology of the network**

- Metropolitan MANETs

- **High Density Areas (HDA): areas with high station density**
- **HDAs can appear and disappear from the network**

- Optimization Problem

- Fine-tune of a broadcasting strategy called DFCN

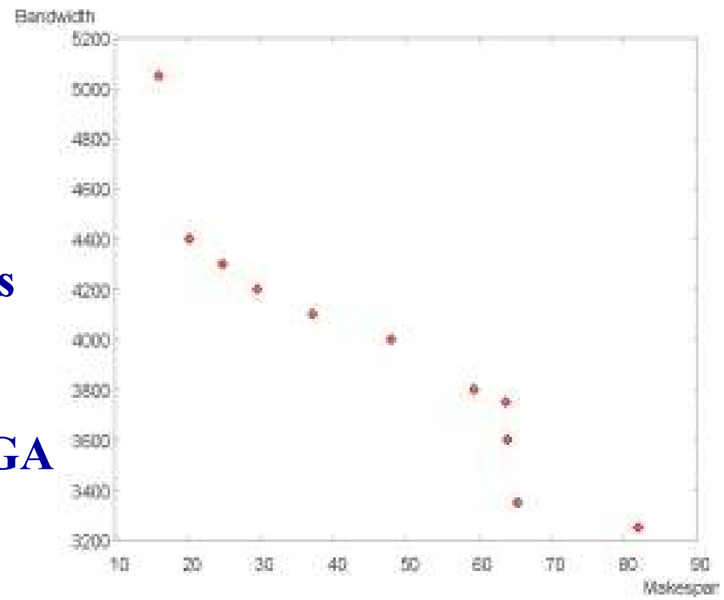
- Target: metropolitan MANETs

- Multiobjective metaheuristics

- EAs: NSGA-II, SPEA2, cMOGA

- Scatter Search: AbYSS

- PSO: MOPSO

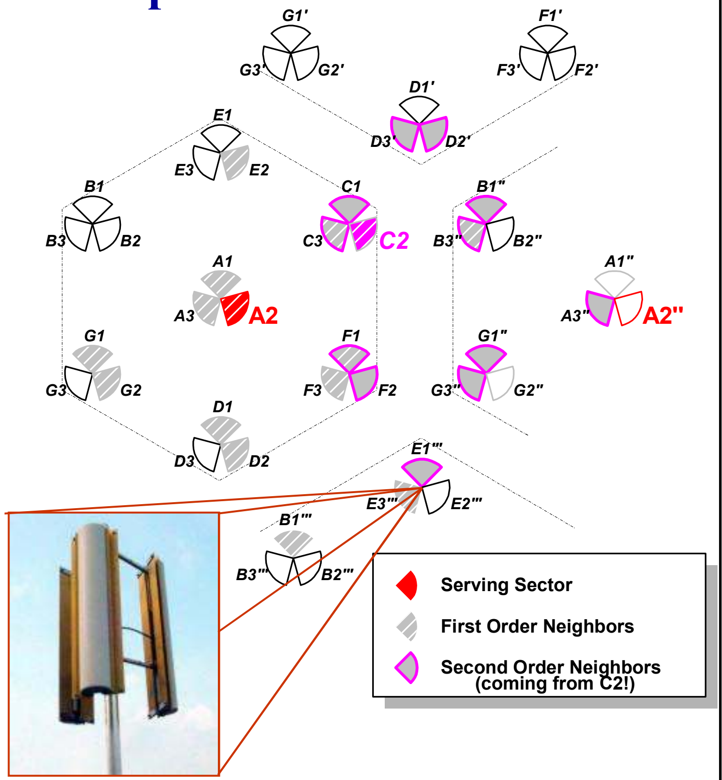


Automatic Frequency Planning in GSM Networks

- Problem definition
  - Allocate frequencies (few dozens) to elementary transceivers (TRXs) of the network (thousands)
  - Frequency reuse is mandatory → this provokes interference → QoS degradation
  - Real world instances of GSM networks currently in use

- GSM architecture
  - Base Transceiver Stations
  - Sectors
  - TRXs

- Metaheuristics used
  - (1,λ) EA, ACO
  - Ongoing:
    - ssGA
    - Grid computing with Condor



Automatic Cell Planning in 2G/3G Networks

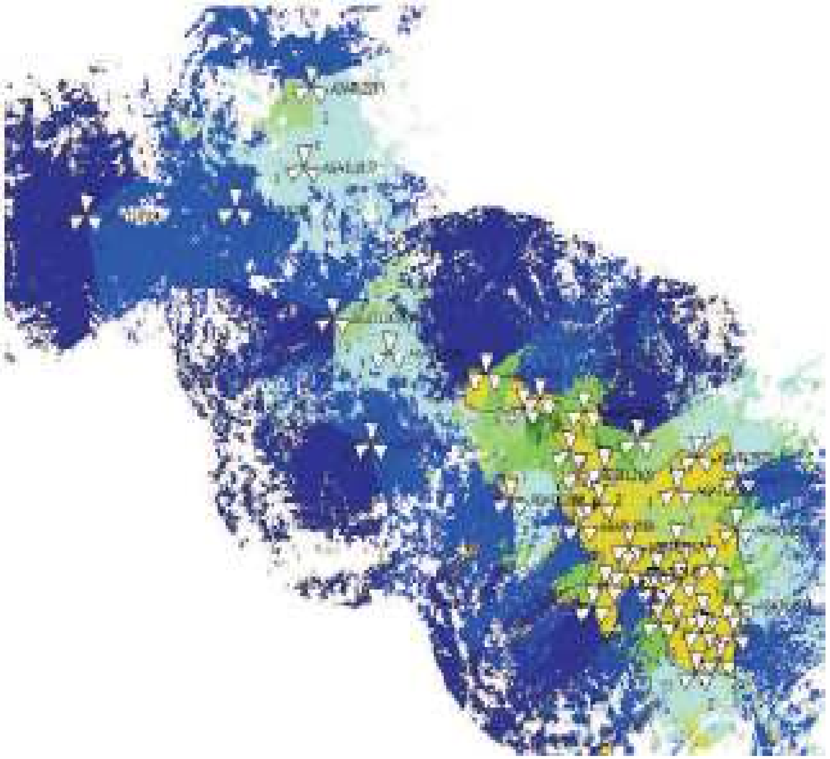
Index

- New Models
- Parallel MetaH
- Hybrid MetaH
- Multiobjective
- ➡ Telecoms.
- Bioinformatics
- Software
- Others

- Problem definition
- **Positioning the sites of the network**
- **Dimensioning a set of parameters for each antenna**
- **Real world instances of cellular networks**
  - **Highly demanding computational costs**

- Objectives
- **Number of sites**
- **Quality of service**
  - **Interferences**
  - **Traffic hold**

- Metaheuristics used
- **- PAES, ssGA**
- **- New work in:**
  - **- AbYSS**
  - **- MOCell**
  - **- Gridifying with Condor/MW**



Radio Network Design: Problem

Index

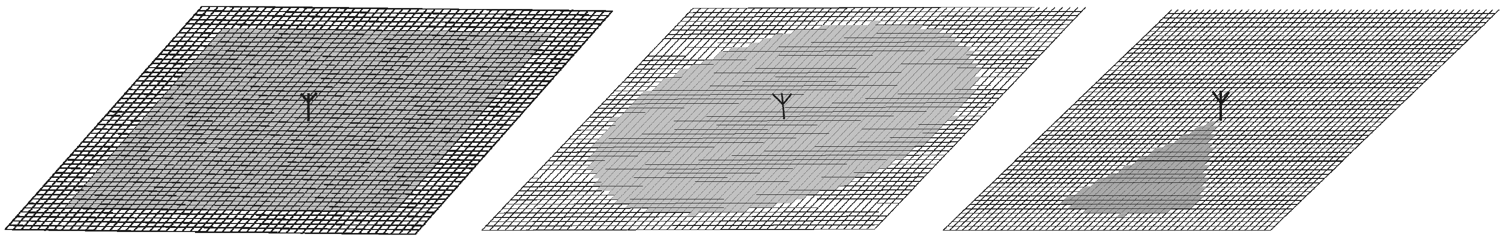
Task: Select geographical locations for cellular antennae from a list of 149 to 349 available locations

Objectives: Maximize the covered area (coverage)
Minimize the number of antennae

Fitness:

$$f(\vec{x}) = \frac{Coverage(\vec{x})^{\alpha}}{Number\ of\ antennae(\vec{x})} \quad \text{with } \alpha = 2$$

Models: Terrain -> Square grid with 287×287 points
Antenna coverage -> Three shapes



Radio Network Design: Algorithms

SA: **Trajectory technique**

Uses mutation to generate S_n from S_a

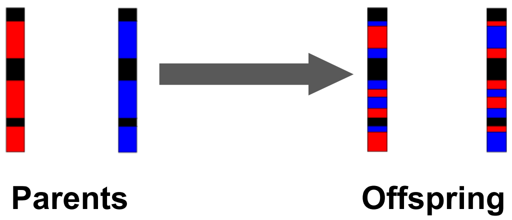
Replaces S_a with S_n with probability *P*

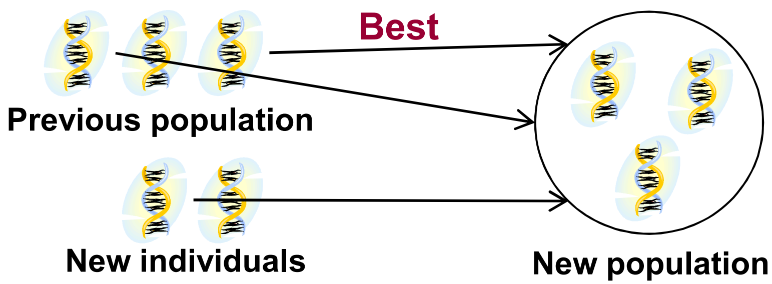
$$P = \frac{2}{1 + e^{\frac{fitness(S_a) - fitness(S_n)}{T}}}$$

CHC: **Evolutionary algorithm**

No mutation, HUX crossover with incest prevention

Elitist selection, Cataclysmic mutation (restarting procedure)



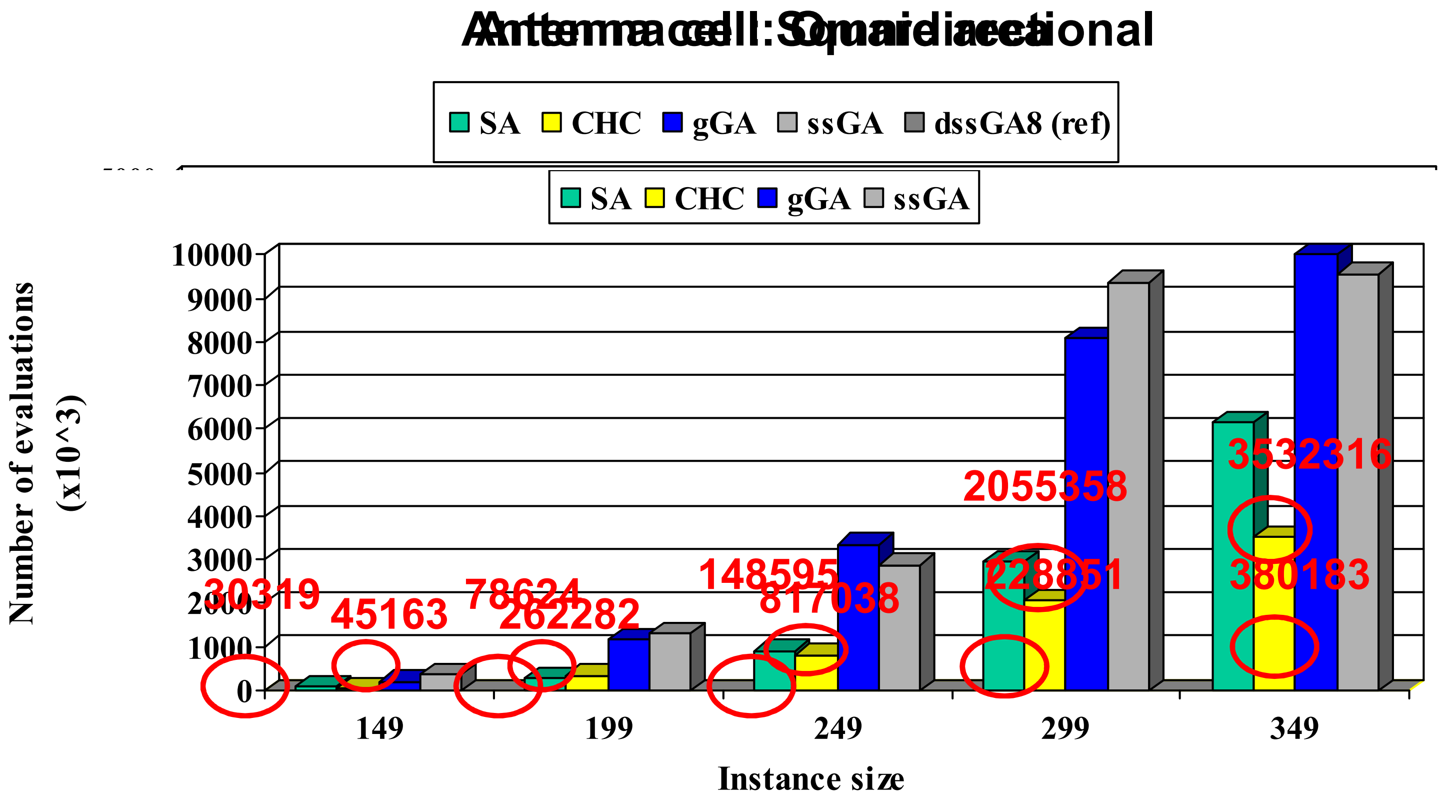


GA: **Evolutionary algorithm**

Mutation, two-point crossover

Selection: Ranking (parents) + tournament (new population)

Radio Network Design: Results



- **CHC** finds the optimal solution with less effort than SA and GA for the five instance sizes using both antenna types
- The geometry of the problem affects the complexity

Wireless Sensor Networks: Problem

Index

Task:

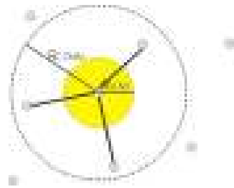
Select geographical locations for sensor nodes
Locations may be chosen from a list, or freely

Objectives:

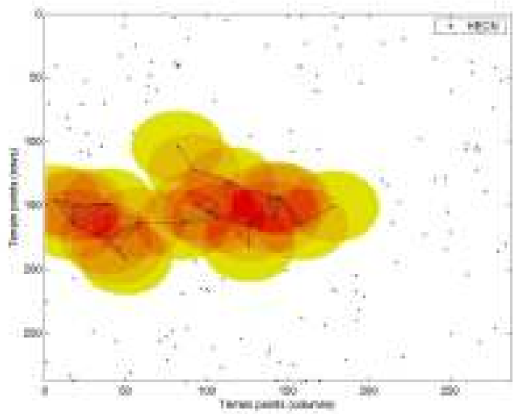
Nodes must form a connected network
Maximize the coverage
Minimize the number of transmissions (energy)

Models:

Node → (RSENS,RCOMM)



Network →

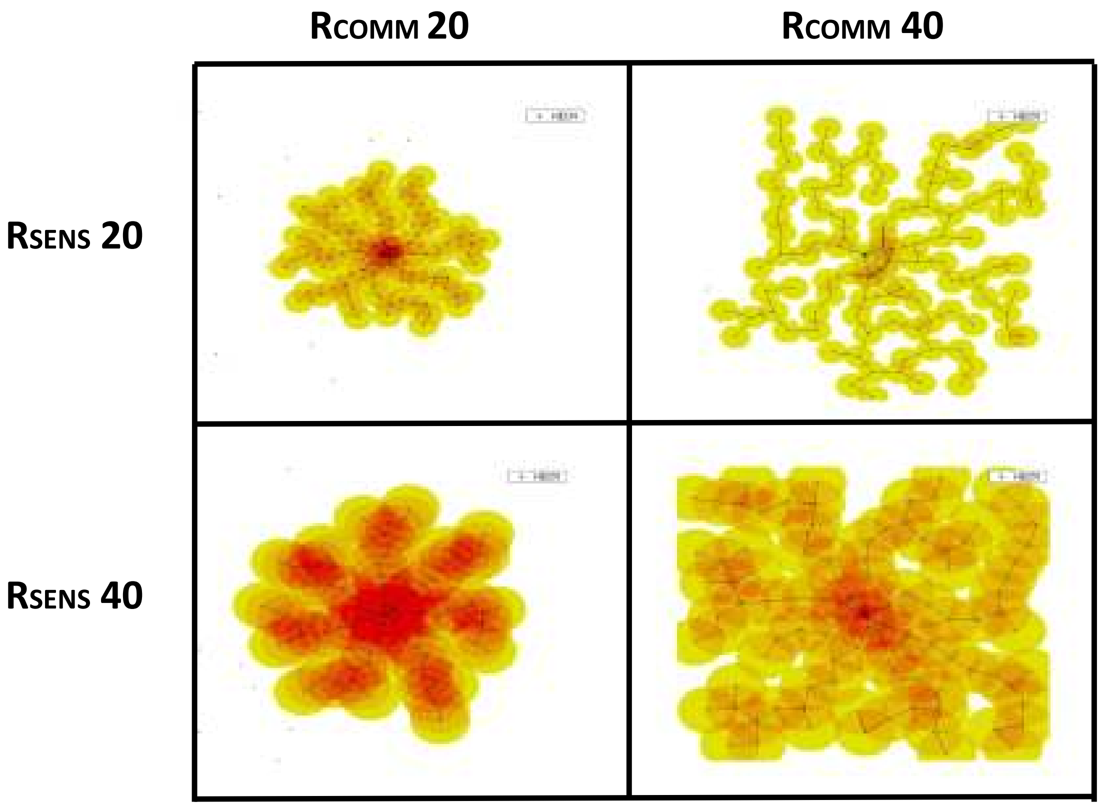


Wireless Sensor Networks: Results

Index

Algorithms: **CHC gets the best results in binary coded instances**
SA gets the best results in real coded instances

Networks: **Network layouts obtained for different sensor nodes**



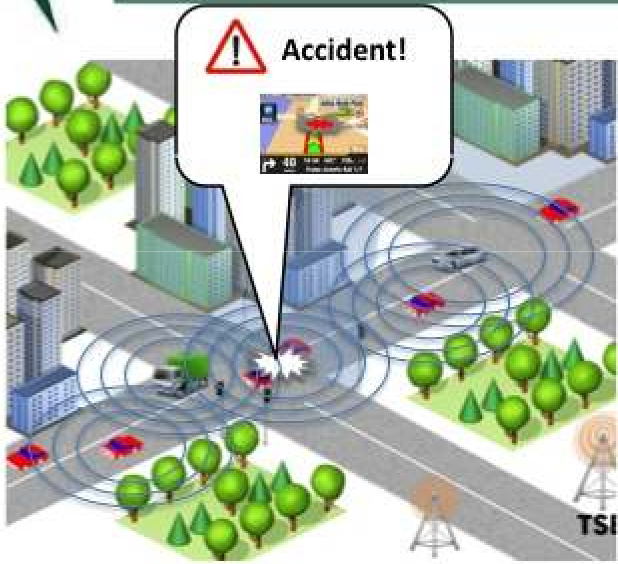
VANETS (I):
Vehicular Ad-hoc Networks

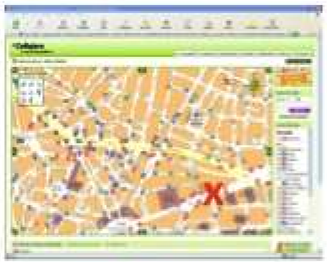
Index

DIRICOM

http://diricom.lcc.uma.es

Information Dissemination





Website

Small Devices

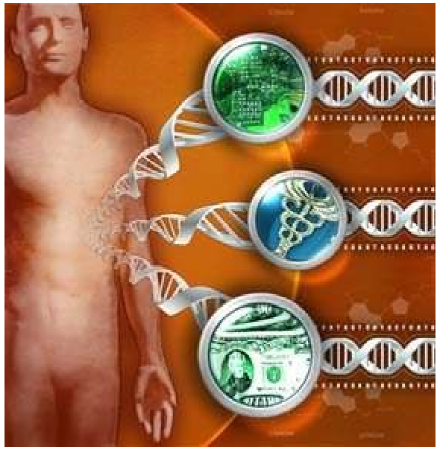
Traffic Control Centre

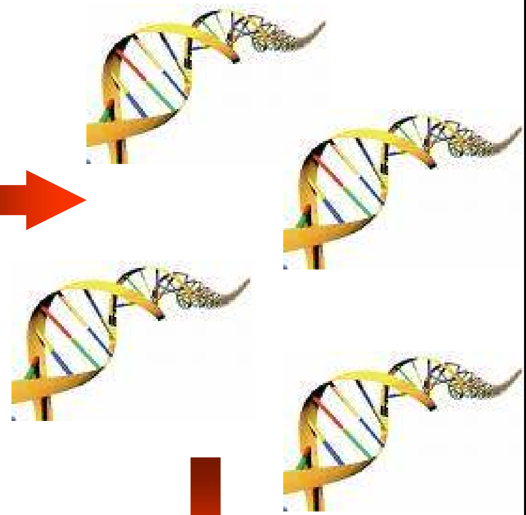
VANETS (II): Vehicular Ad-hoc Networks

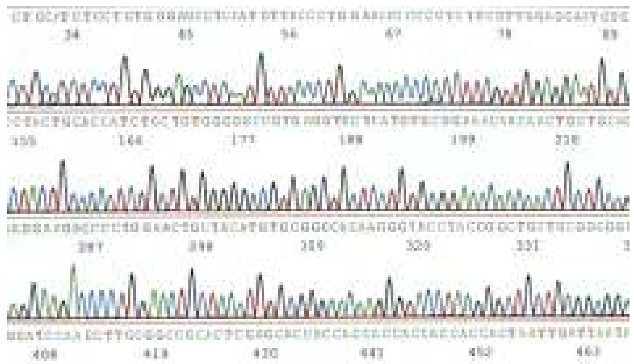
Index

Index

Bioinformatics: Fragment Asembly Problem (I)







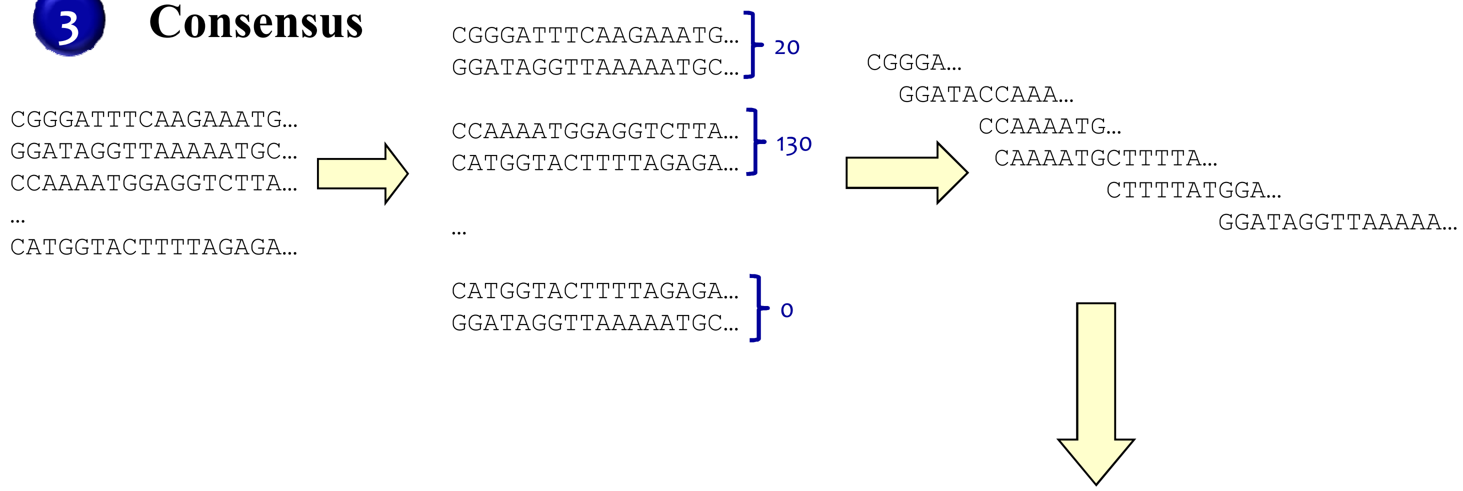
Input to the assembler:

CGGGATTTCAAGAAATG...
GGATAGGTTAAAAATGC...
CCAAAATGGAGGTCTTA...
...
CATGGTACTTTTAGAGA...

Bioinformatics: Fragment Asembly Problem (II)

The main phases performed by assemblers are:

1. **Overlap**
2. **Layout**
3. **Consensus**



Output of Assemblers: CGGGATACCAAATGCTTTTATGGATAGGTTAAAA ...

Our approaches:

- **Parallel metaheuristics**
- **Specialized heuristics (PALS)**

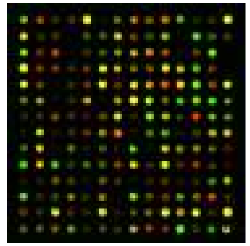
Our results:

- **Able to solve very large sequences**
- **Large efficacy and efficiency**

Bioinformatics: Gene Selection (I)

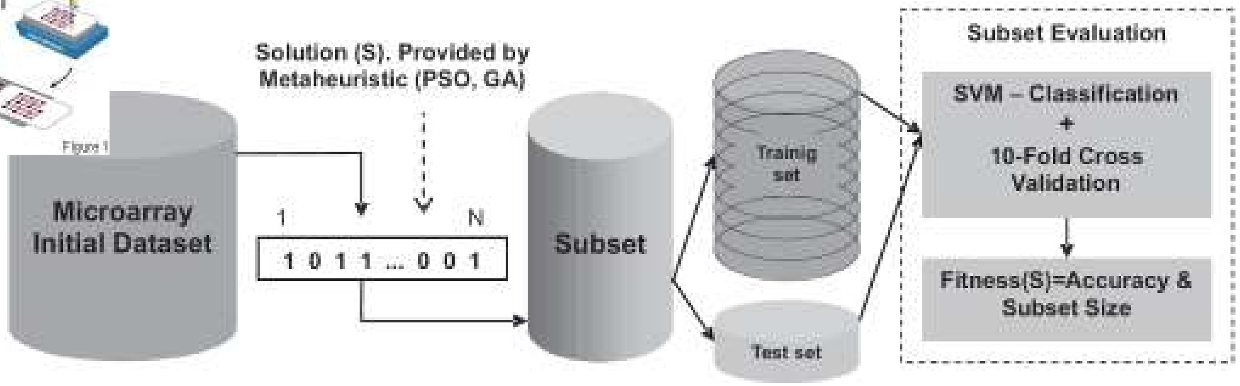
Problem: Gene selection and cancer classification of DNA Microarray, Feature selection definition:

$$F' = argmax_{G \subset \Gamma}\{\Theta(G)\}$$



Objectives:
- Maximize accuracy of prediction
- Minimize the number of selected genes
- Maximize ROC factors (sensibility and specificity)

Phases: Feature selection, training, validation, fitness calculation



Bioinformatics: Gene Selection (II)

Index

Fitness: **Monobjective: aggregative**

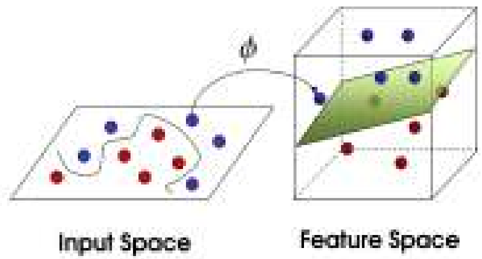
(alpha*100/accuracy + beta * #features)

Multiobjective:

- **2 objs (*accuracy, #features*)**
- **3 objs (*sensibility, specificity, #features*)**

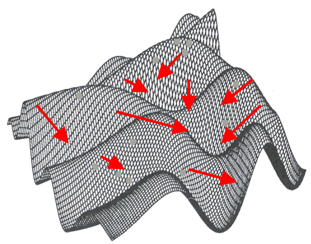
Classification: **2 Classifiers:**

- **Support Vector Machines**
- ***K*-Nearest Neighbors**



Validation: **Cross-validation.**

- **Leave One Out CV**
- **10-fold CV**



Algorithms: **Metaheuristics**

- **Binary Geometric PSO for feature selection**
- **GA with SSOCF crossover for feature selection**

Bioinformatics: Gene Selection (III)

Instances: **Available large scale datasets of well-known cancer DNA Microarrays: Leukemia AML-ALL, Colon, Prostate, Lung, Ovarian, Breast (e.g. breast 24481 genes and 78 patient samples)**

Results: **Comparison against other techniques in the literature**

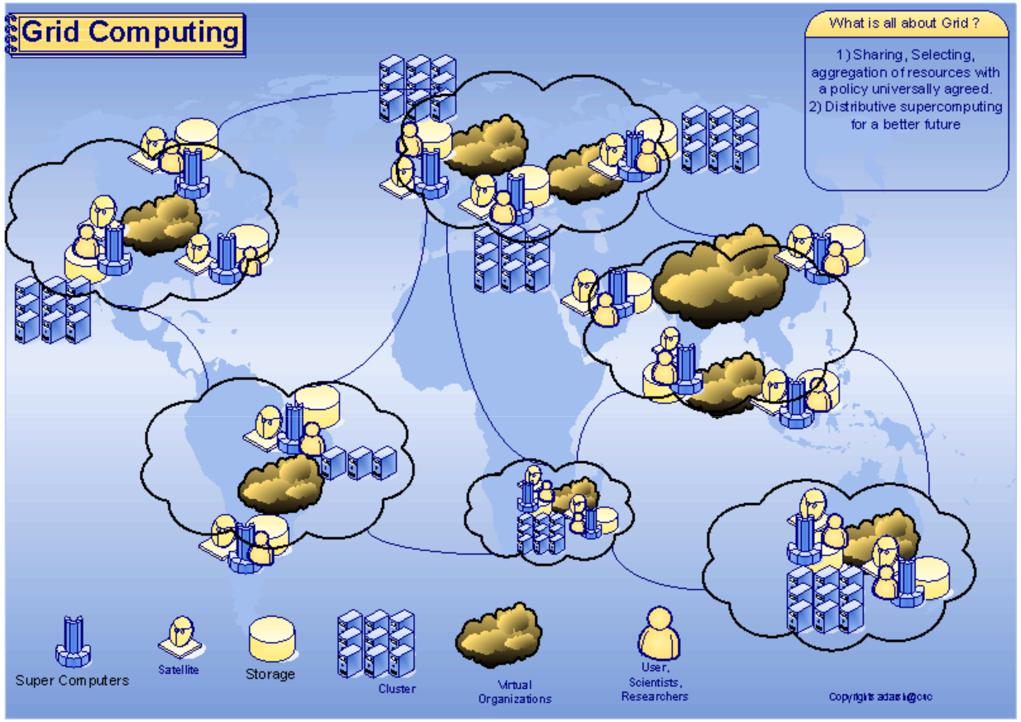
Dataset	PSO	GA	Huerta et al.	Juliusdottir et al.	Deb et al.	Guyon et al.	Yu et al.	Liu et al.	Shen et al.
Leukemia	99.33(3)	99.79(4)	100(25)	-	100(4)	**100(2)**	87.44(4)	-	-
Breast	**90.64(4)**	89.02(4)	-	-	-	-	79.38(67)	-	-
Colon	**100(3)**	100(4)	99.41(10)	94.12(37)	97(7)	98(4)	93.55(4)	85.48(-)	94(4)
Lung	100(4)	**97.38(3)**	-	-	-	-	98.34(6)	-	-
Ovarian	100(4)	**100(3)**	-	-	-	-	-	99.21(75)	-
Prostate	**100(4)**	**100(4)**	-	88.88(20)	-	-	-	-	-

Leukemia Gene Subsets

- **By PSOsvm**: K01383 *at, U03056 at, J04130 s at*
- **By GAsvm:** *L40379 at, S85963 at, U83192 at, Z49099 at*

Bioinformatics and Grid Computing

Index



Condor
High Throughput Computing

At present:	**using a stable grid with 300 computers at UMA**
In progress:	**stable grid of 600 computers at UMA**
Next step:	**connect to global grids in Europe**

Finding Safety Property Violations in Concurrent Systems (I)

- **Objective:** **find a counterexample for a safety property in a concurrent model**

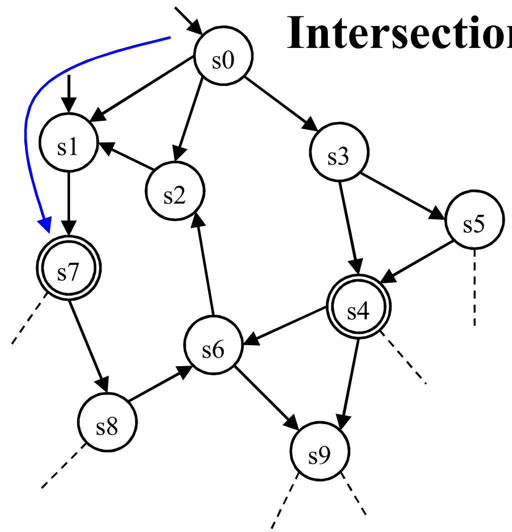
- **Safety properties are those expressed by an LTL formula of the form:**

$$f = \Box p$$

where *p* is a past formula (with only past operators)

- **Finding one counterexample ≡ finding one accepting state in the intersection Büchi automaton (graph exploration problem)**

Intersection automaton



Safety Properties

Deadlocks

Invariants

Assertions

…

Index

New Models

Parallel MetaH

Hybrid MetaH

Multiobjective

Telecoms.

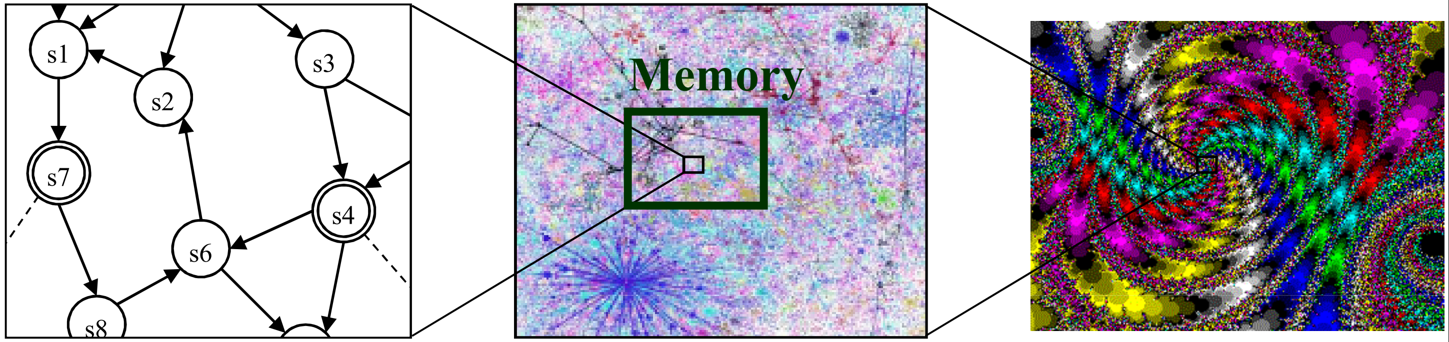
Bioinformatics

Software

Others

Finding Safety Property Violations in Concurrent Systems (II)

- Number of states very large even for small models



- For example: Dijkstra Dining Philosophers

- *n* philosophers → 3^n states
- 20 philosophers → 1039 GB for storing the states

Finding Safety Property Violations in Concurrent Systems (III)

Index

- ACOhg is a new Ant Colony Optimization model that can be applied to optimization problems with an unknown and/or very large construction graph

Who can really find errors?

Models	BFS	DFS	A*	BF	ACOhg
giop22		●	●	●	🐜
needham	●	●	●	●	🐜
phi16			●	●	🐜
pots	●	●	●	●	🐜
marriers4				●	🐜
marriers20					🐜

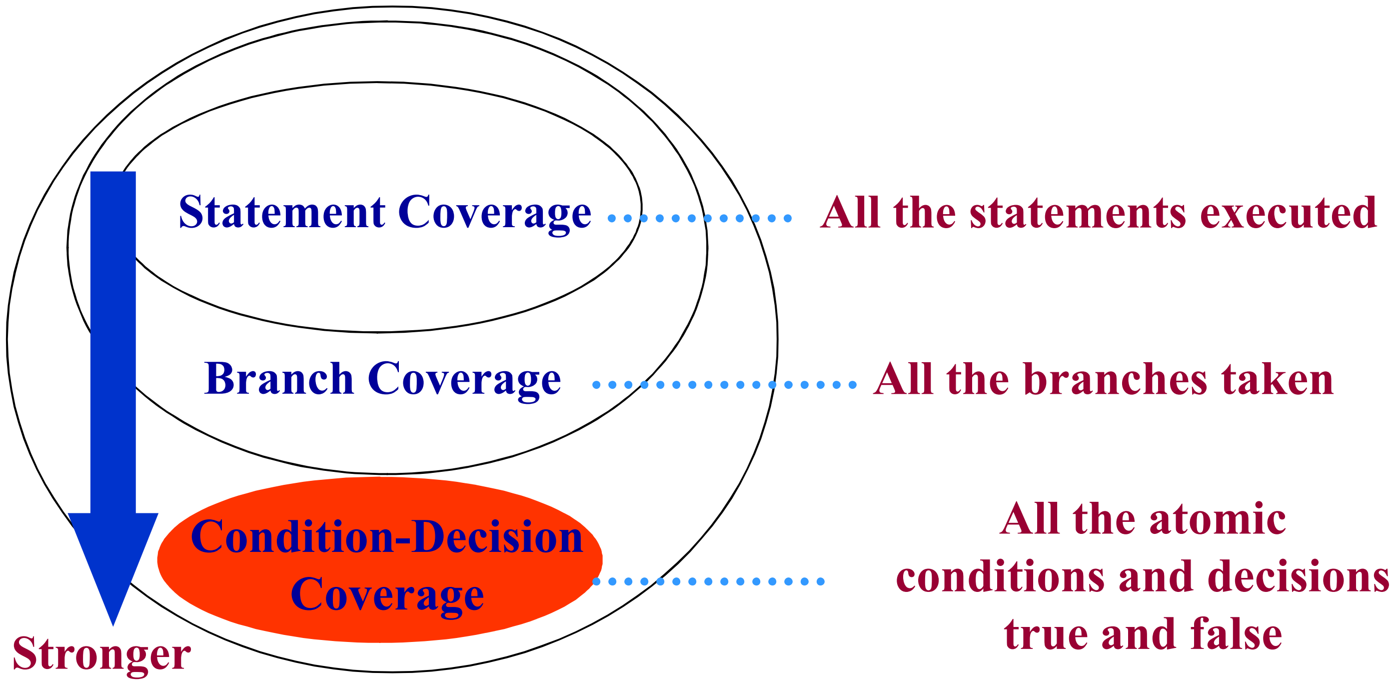
- ACOhg is a very robust algorithm for this problem and it outperforms traditional algorithm from the model checking domain

Software Testing (I)

- Objective: propose a good set of test cases for a program

- The previous objective is too fuzzy, one concrete objective is: find a test case set fulfilling a test adequacy criterion

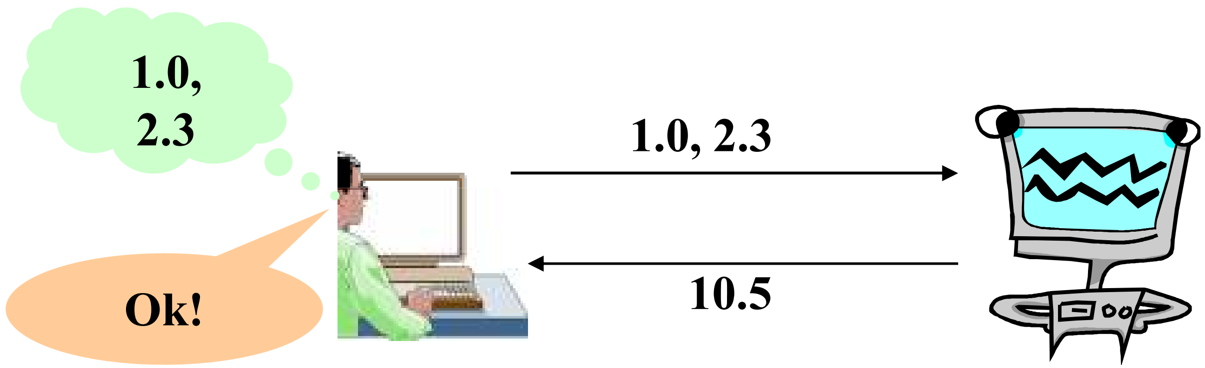
- Some examples of test adequacy criterion are:



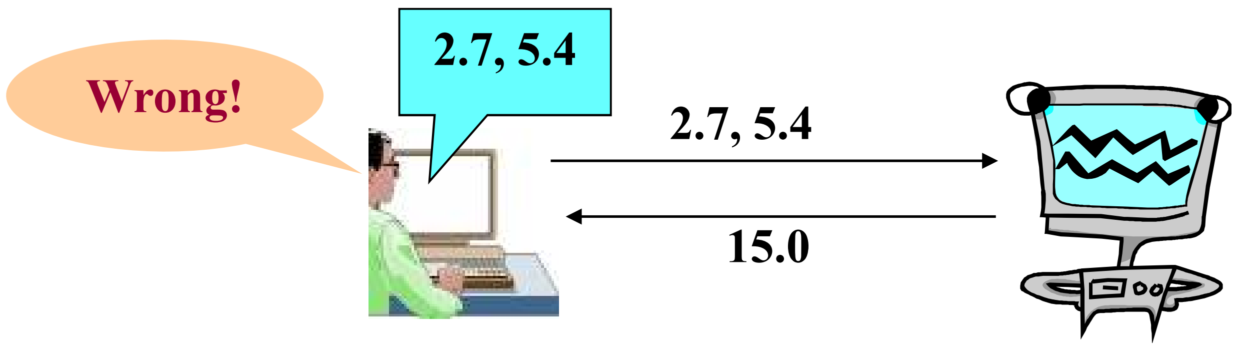
Index

Software Testing (II)

- **After codification, software products require a test phase**
- **The objective is to find errors and to ensure software correctness**
- **Software companies dedicate 50% of resources to this task**



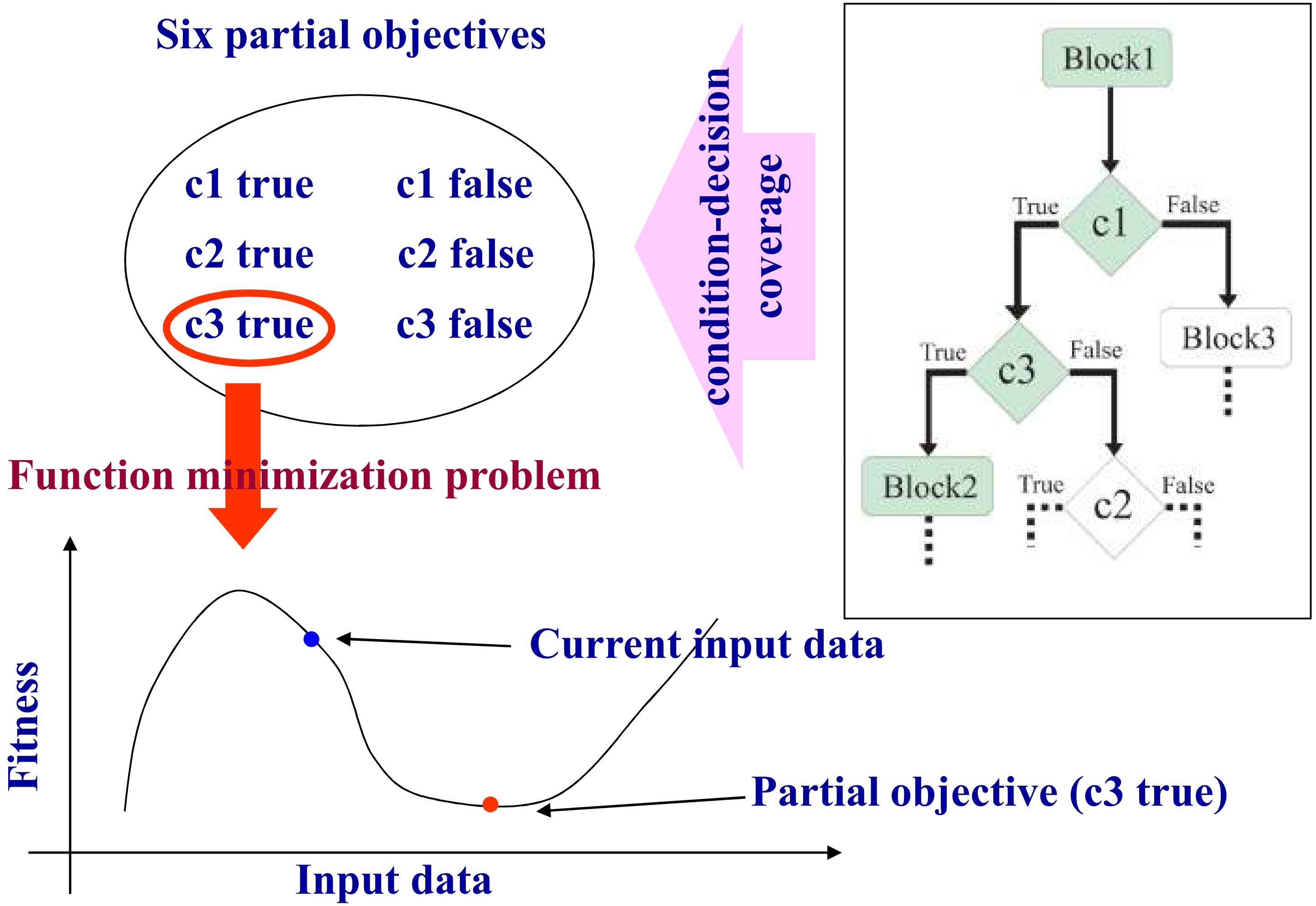
- **We propose an automatic tool to generate the input data for the test**



Index

Software Testing (III)

- The global objective is broken down in small partial objectives



Index

Software Testing (IV)

- Some results with PSO, ES, and GA (corrected condition coverage)

Programs	PSO		ES		GA	
	Cov.	Evals.	Cov.	Evals.	Cov.	Evals.
triangle	93.98	11295.77	**99.84**	**2370.03**	99.67	3209.47
calday	**100.00**	179.33	98.18	3166.47	90.91	**75.03**
select	**88.89**	380.13	83.33	**13.27**	83.33	83.20
bessel	**97.56**	**116.90**	**97.56**	350.63	**97.56**	533.03
sa	**100.00**	**165.67**	99.94	2337.30	96.72	176.63
netflow	97.77	4681.70	**98.17**	**307.77**	96.42	917.90

- PSO and ES have similar efficacy

- The coverage obtained by GA is always reached or outperformed by PSO or ES in all the cases

Index

New Models

Parallel MetaH

Hybrid MetaH

Multiobjective

Telecoms.

Bioinformatics

Software

Others

Others: Strip Packing (I)

- *N* rectangular pieces p_i with a height h_i and a width w_i and a rectangular container (the strip) with width *W* and unbounded height.
- **Objective**: To allocate all the pieces into the strip
  - without overlaping,
  - without rotating,
  - with their edges parallel to the edges of the strip,
  - Bottom-up, minimizing the height of the used strip.

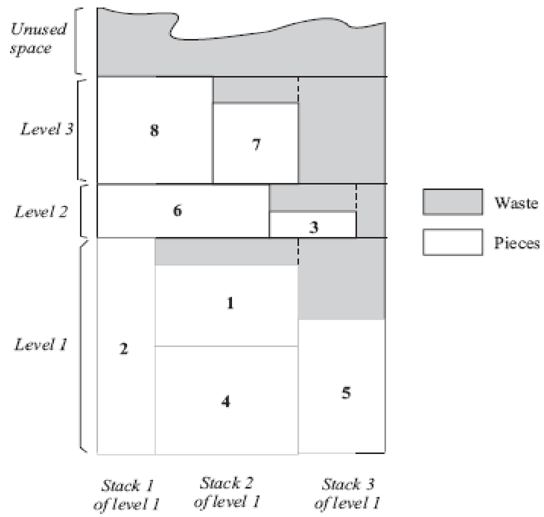
  (Eq: To find a packing pattern that fulfils all these requirements)
- **Restriction**: three-stage guillotine patterns.
- **Scientific interest**: NP-hard problem.
- **Applications**: Paper, cloth, wood, and glass industries.

Representation

Chromosomes: sequences (permutations) of pieces which define the input for a layout algorithm.

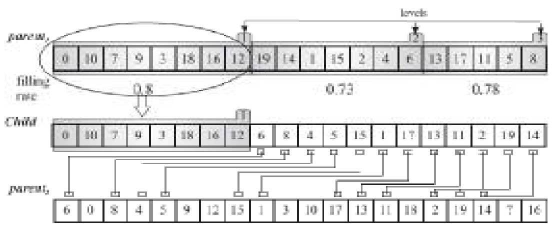
Layout algorithm: a next fit heuristic that generates three-stage guillotine patterns.

Fitness function: $$F(\pi) = strip.length - \frac{l.waste}{l.area}$$



Others: Strip Packing (II)

Evolution Step



Best Inherited Level Recombination: Transmits the levels with the highest filling rate from one parent to the child.

Mutation: Best and Worst Stripe Exchange (BW_SE). Pieces of the best level are allocated in the first positions while the pieces of the worst level are asigned to the last positions.

Adjustment Operator: Applies a First Fit heuristic and the obtained layout is codified in a chromosome.

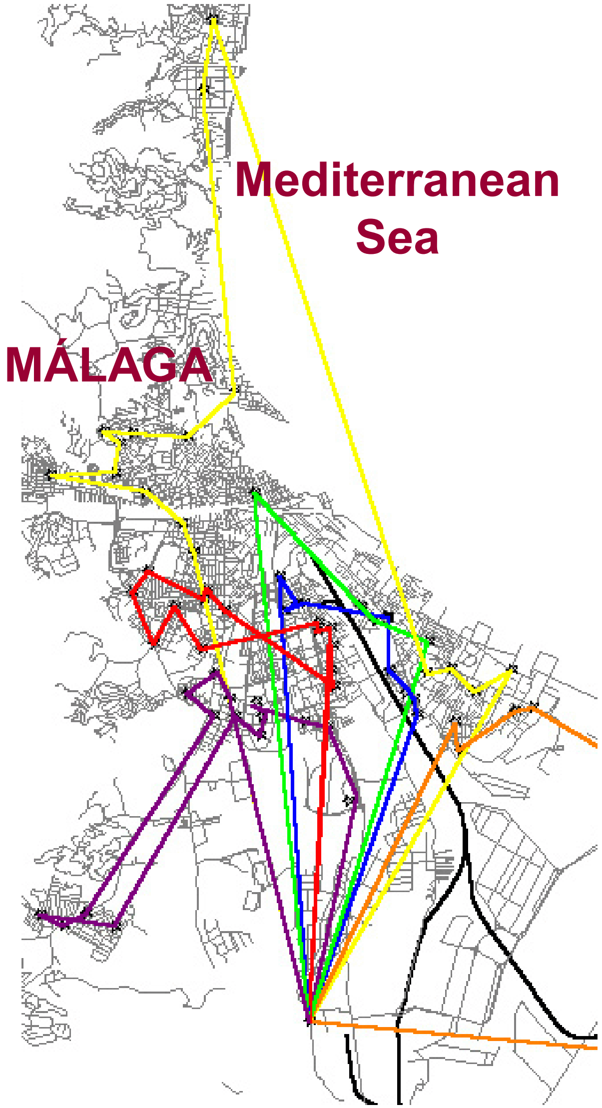
Initial Seeding

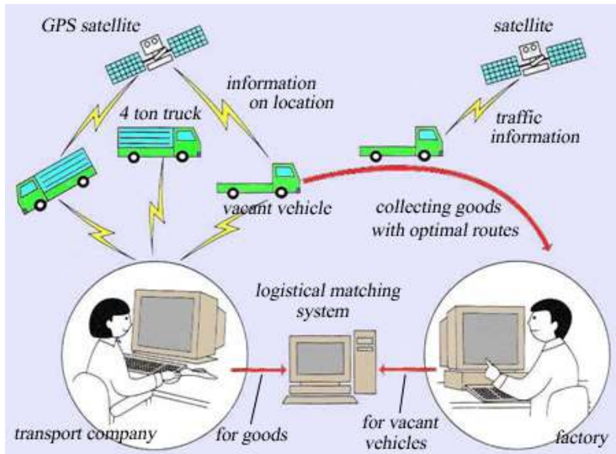
#	Rule Description	#	Rule Description
1	sorts pieces by decreasing width.	2	sorts pieces by increasing width.
3	sorts pieces by decreasing length.	4	sorts pieces by increasing length.
5	sorts pieces by decreasing area.	6	sorts pieces by increasing area.

#	Rule Description
7	sorts pieces by alternating between decreasing width and height.
8	sorts pieces by alternating between decreasing width and increasing height.
9	sorts pieces by alternating between increasing width and height.
10	sorts pieces by alternating between increasing width and decreasing height.
11	the pieces are reorganized following the BFDH heuristic.
12	the pieces are reorganized following the FFDH heuristic.
13	The packing pattern remains without modifications, so here the rule preserves the original piece position (random generation).

Others: Logistics

Index





Others: Dynamic Problems

Index

Optimal location of aerogenerators

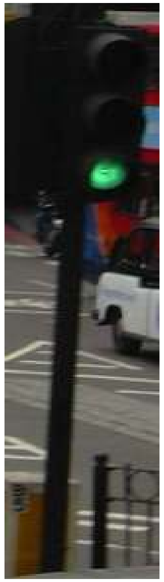
Wind farm design

Aerogenerator design

Others: Dynamic Problems

DYNAMIC OPTIMIZATION

Traffic Light Controllers



Elevator Systems

Gas Engine Control

End of Presentation

Málaga

http://neo.lcc.uma.es