MATHEURISTICS FOR MANAGING OPERATIONAL PROBLEMS OF LARGE SIZE

Éric Taillard

HEIG-VD, University of Applied Sciences of Western Switzerland

Dealing with large problem instances

- Decompose into smaller part
- k-mean algorithm
- Improvement of k-mean algorithm

Matheuristic

- Fix and optimize framework
- POPMUSIC framework

Applications of POPMUSIC

- p-median problem
- Map labelling
- Location routing

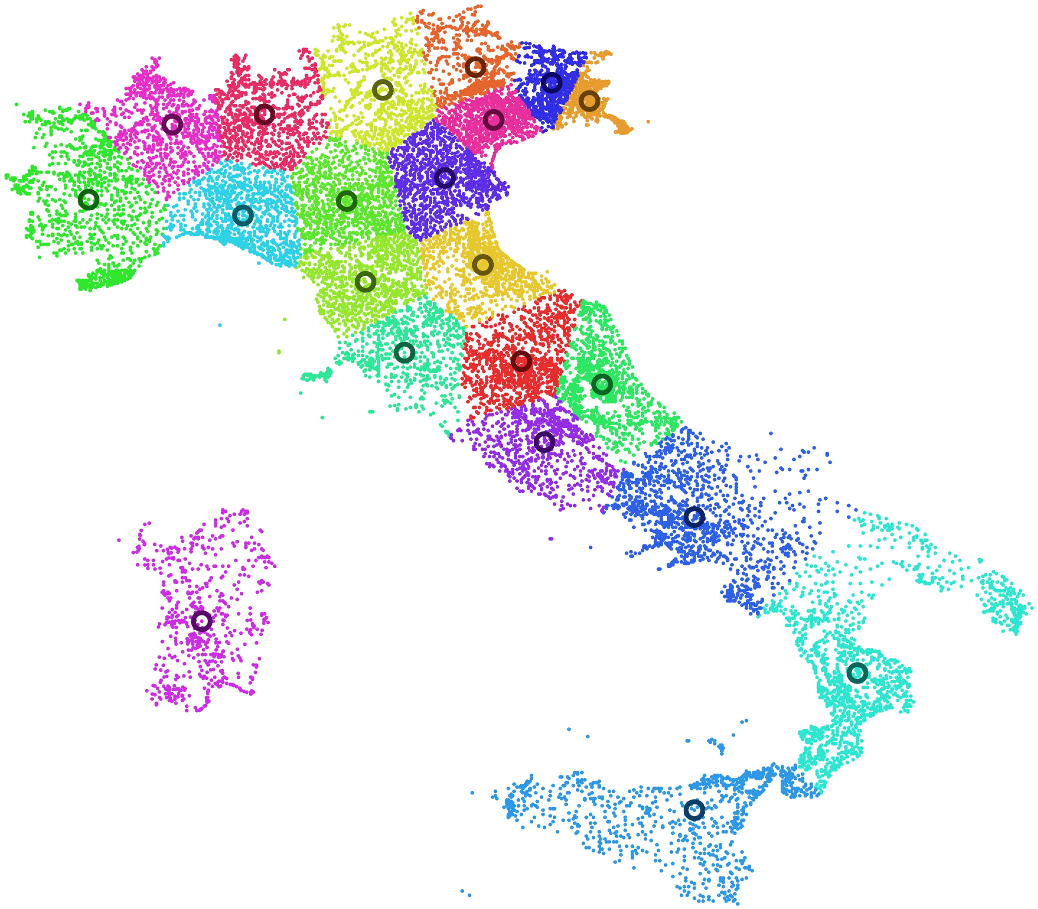
DECOMPOSITION INTO SMALL PARTS : THE P-MEDIAN PROBLEM

Given :

n elements $\in I$ with dissimilarity (distance) measure $d(i, j)$ between them

Find :

p central elements $\{c_1, \ldots, c_p\} \in I$ minimizing $\sum_{i=1}^{n} min_{j=1,\ldots,p} d(i, c_j)$



K-MEAN (P-MEDIAN) ALGORITHM

1. Assign a (random) position to the *p* central elements

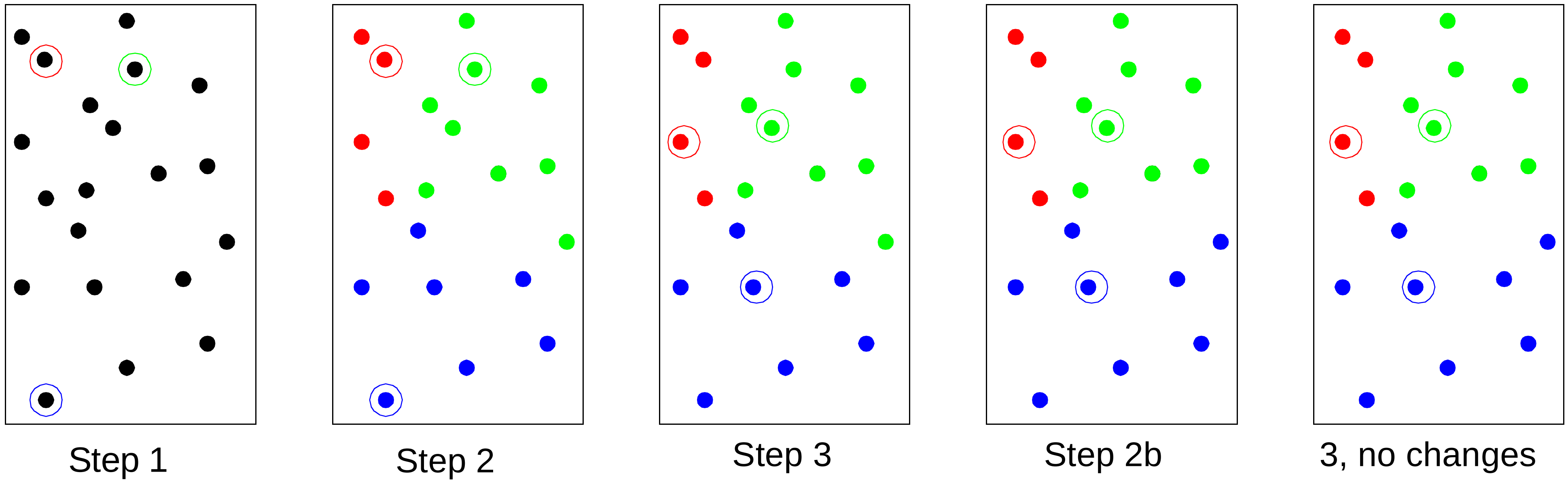
   Repeat

   2. Assign each element to its closest centre (build clusters)

      For each cluster

      3. Relocate the central element at best position for the cluster

4. While clusters change



Fast algorithm : O($p \cdot n + n^2/p$) for p-median, O($k \cdot n$) for k-mean

May produce bad quality solutions if *p* higher than few dozens

Goal :

Decomposing a set $E = \{1, \ldots, n\}$ into p clusters $C_1, \ldots, C_p$ with $\sim n/p$ elements each

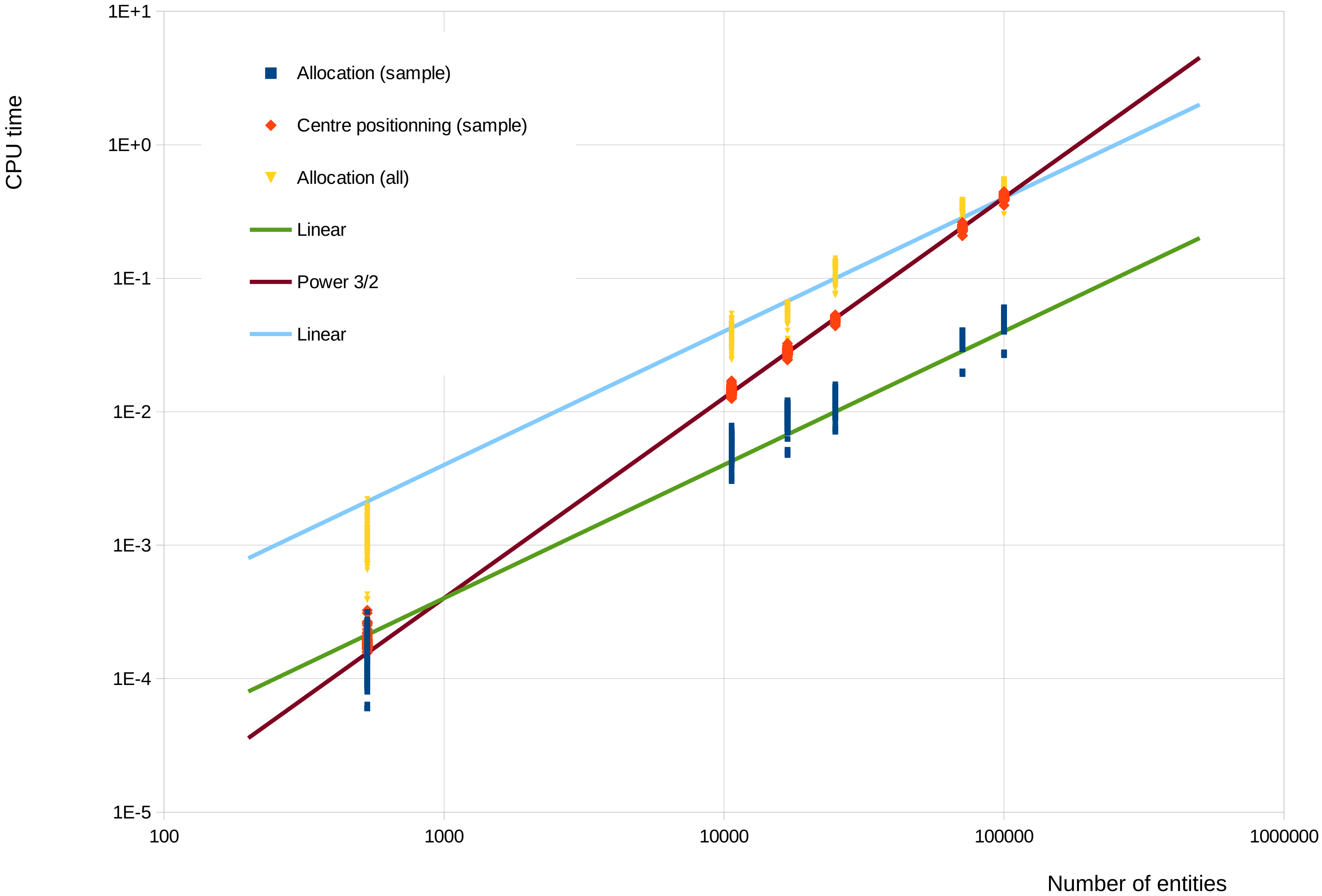
```
Randomly select a sample of e = min(n, 30·p) elements among the n entities
Randomly choose p centres c_1, …, c_p among the e entities
f = 0.6;
λ_j = 0; j = 1, …, p; // Penalty for each centre
for 30 iterations do
    for i = 1 to e do
        Allocate entity i to the centre j minimizing d(i,j) + λ_j
    for j = 1, …, p do
        Find the best position of centre c_j among entities assigned to it
    Compute solution cost C and store solution if best improved
    f ← 0.99 · f
    for j = 1, …, p do
        λ_j ← Max(0, λ_j + f · C · (n_j – e/p)·e^2)
        // n_j : number of entities allocated to centre j
for i = 1 to n do
    Allocate entity i to the centre j of best solution minimizing d(i,j)
```

Complexity

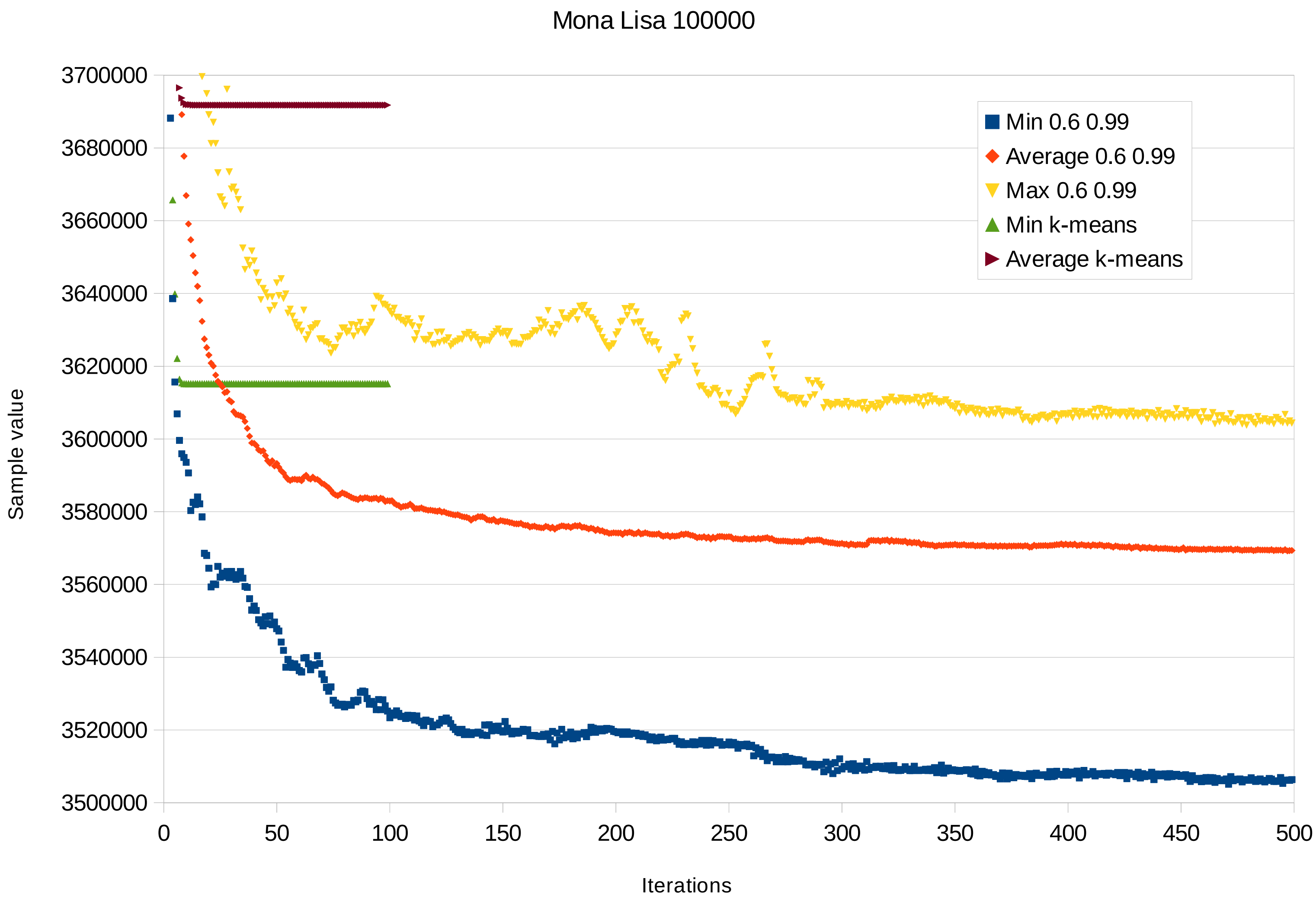
$$\Theta(e \cdot p + e^2 + n \cdot p) \Rightarrow \Theta(n^{3/2}) \text{ if } p \text{ in } \Theta(\sqrt{n})$$

CPU Time

Seconds per iteration on intel i7 930 2.6GHz

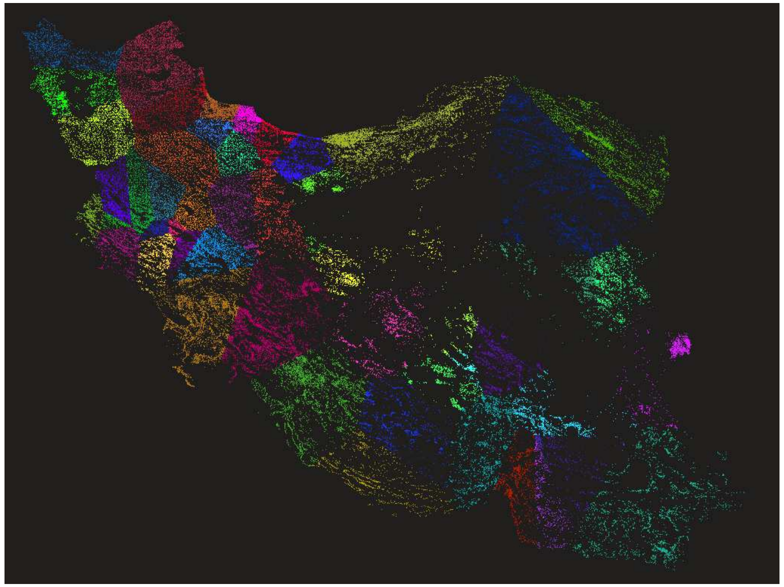


Mona Lisa (100,000 elements), 30 independent runs

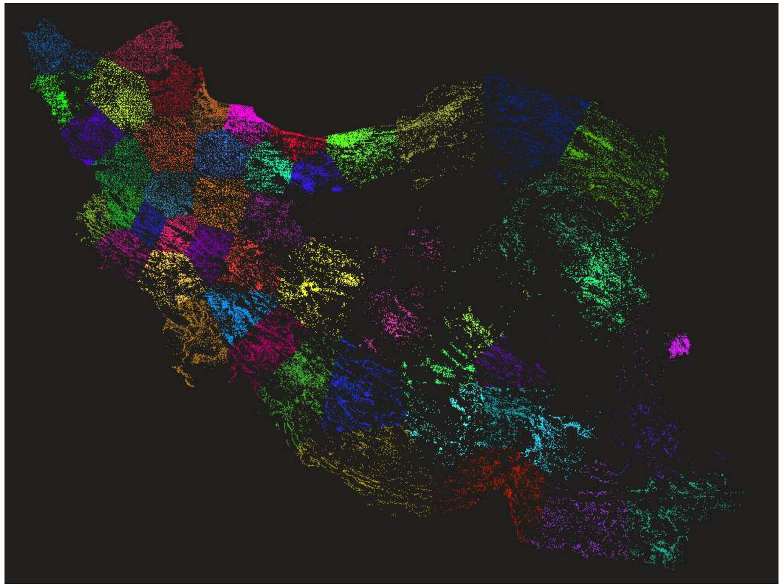


APPLICATION OF THE ALGORITHM

63551 entities

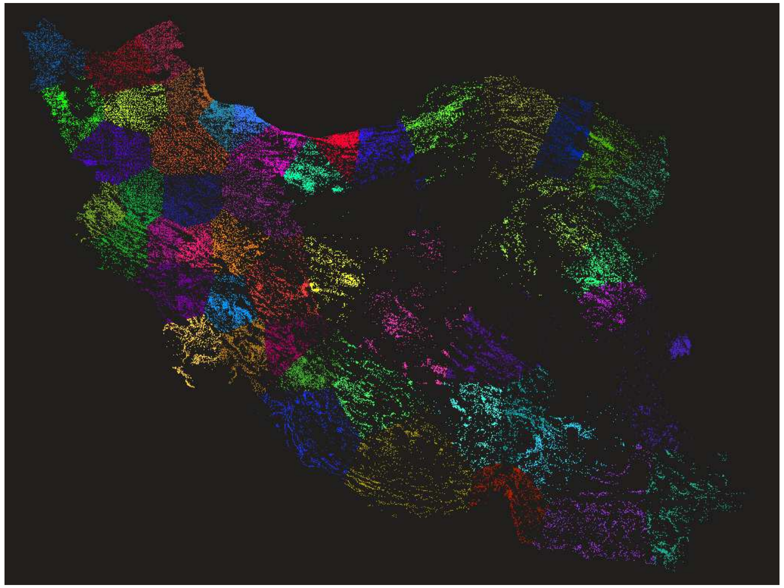


Initial random solution

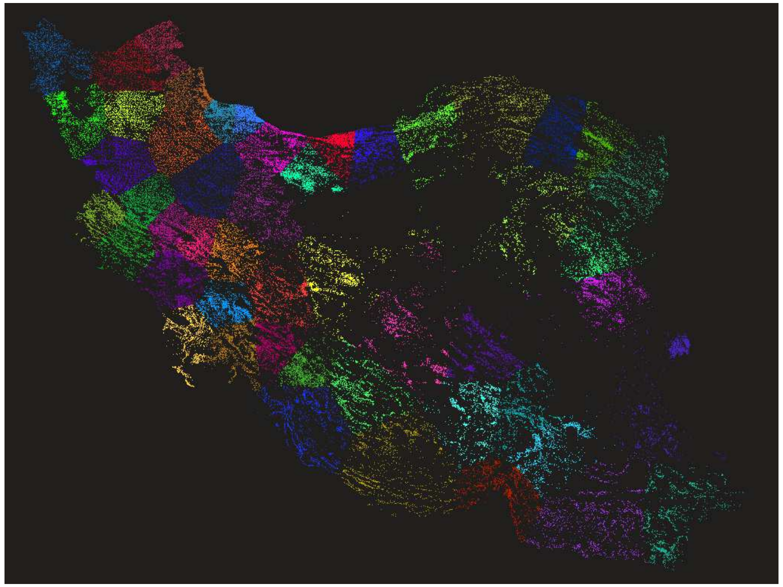


k-median algorithm (34 iter.)

Modified p-median, iter. 34



Final, modified p-median (124 it.)



Pertinence of problem decomposition

Hypothesis

Large problem instances but moderate dimension

⇒ 2 elements close to a third one are also close

⇒ 2 elements far away cannot be both close to a third one

Distant elements are not directly connected together in reasonable solutions

Reasonable solutions are composed of sets with about *C* elements (independent of problem size)

Example : The number of letters a postman can deliver in a day does not depend on the total number of people living in the country

Efficient decomposition into *n*/*C* clusters

Bi-level decomposition :

Decompose n entities into $\sqrt{n}$ cluster with about $\sqrt{n}$ entities in $O(n^{3/2})$

Decompose each of the $\sqrt{n}$ clusters into $\sqrt{n}/C$ clusters with about C elements in $O(n^{3/2})$

Clustering allows to build efficiently solutions of moderate quality to large instances

Candidate list, strongly determined and consistent variables (Glover)

Large neighbourhoods (Shaw)

Exchange (Pochet & Wolsey)

Fix-and-Optimize (Helber & Sahling)

While stopping condition not met **loop**
   Select r variables of the problem
   Tentatively optimize the solution by modifying only the r selected variables

Weakness of the approach

"Endless" loop $\binom{n}{r}$ different possibilities of selecting variables

Most of the possibilities are not pertinent

POPMUSIC idea

Built only pertinent sub-problems

Natural stopping criterion

Part :

Elements belonging to a cluster

Distance :

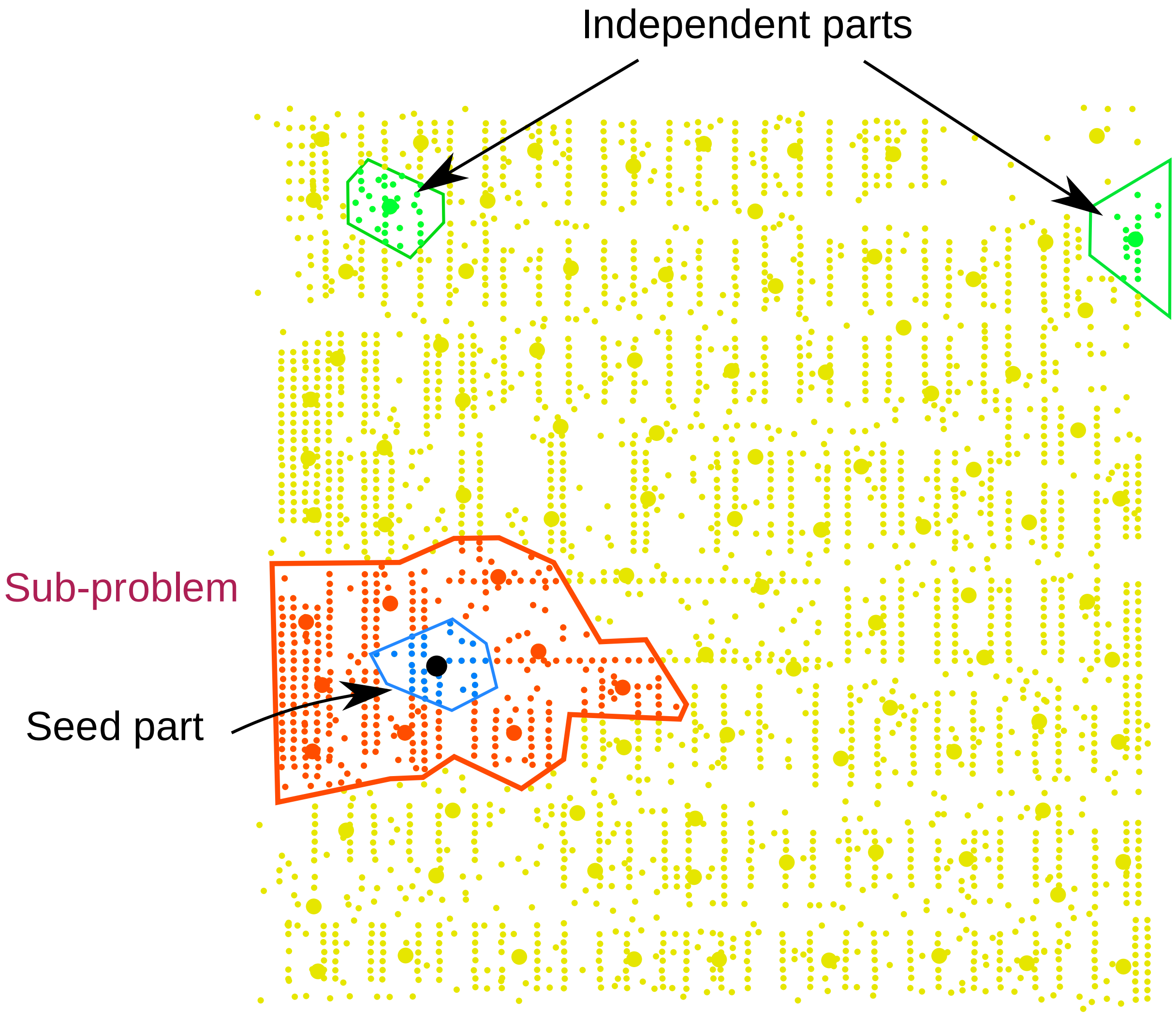
Average dissimilarity between elements of different groups,

Distance between centres

Optimization process :

Exact algorithm or

Modified p-median



Input

Solution $S = s_1 \cup s_2 \cup \ldots \cup s_p$ // p disjoint parts

$O = \emptyset$ // Set of " optimized " seed parts

While $O \neq S$, **repeat** // Parts may still be used for creating sub-problems

1. Choose a seed part $s_i \notin O$
2. Create a sub-problem R composed of the r " closest " parts $\in S$ from s_i // r : parameter
3. Optimize sub-problem R
4. **If** R improved **then**
   Set $O \leftarrow O \setminus R$
   Else
   Set $O \leftarrow O \cup s_i$

POPMUSIC CHOICES

How to get an initial solution

Balanced clustering

Definition of a part

Distance between two parts

Seed part choice

Random, O managed as a stack, …

Parameter *r*

Depends on optimization procedure capability

Optimization procedure

Exact method, matheuristic, metaheuristic

Variants :

Slower :

set $O \leftarrow \emptyset$ instead of set $O \leftarrow O \setminus R$

Faster :

set $O \leftarrow O \cup R$ instead of set $O \leftarrow O \cup s_i$

Capacitated Vehicle Routing Problem

Given :

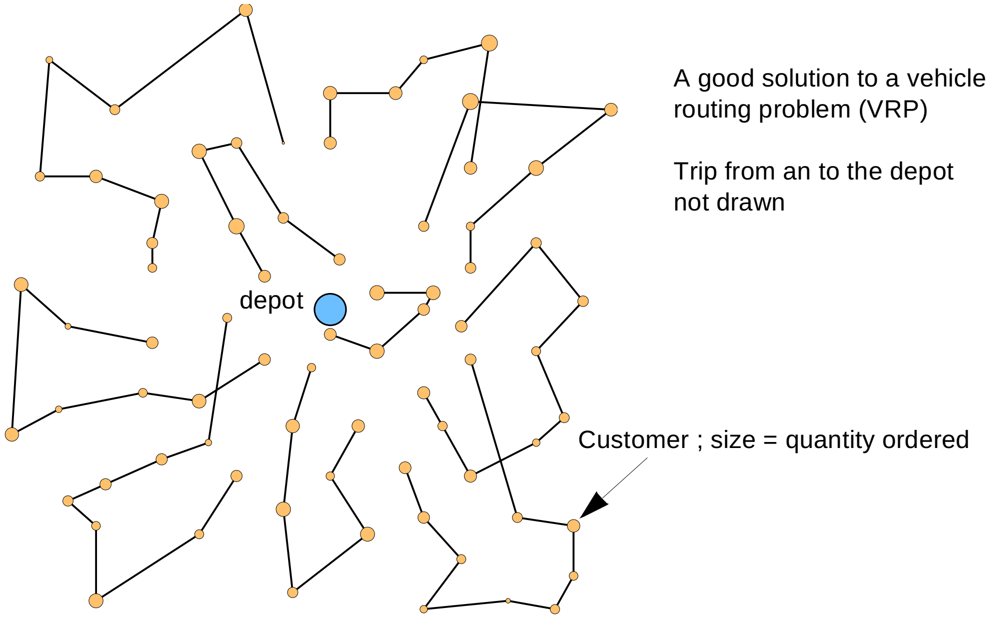
n customers and 1 depot

q_i : quantity ordered by customer i

Distances between each pair of customer

and between depot and customer

Q : vehicle capacity



Find :

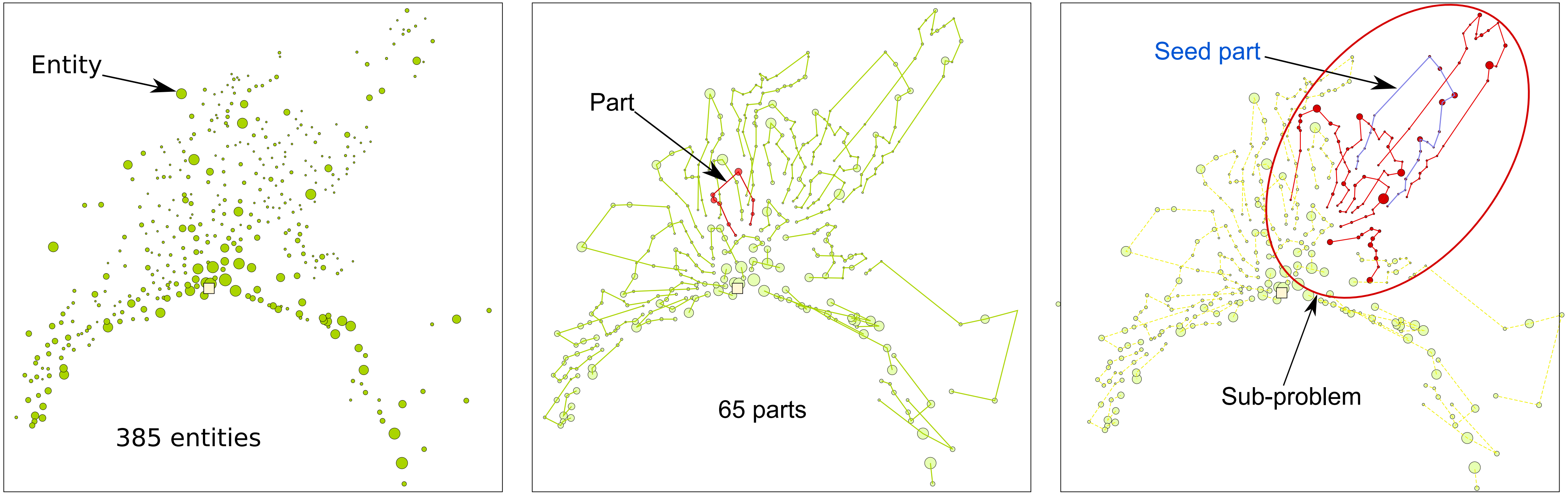
Set of tours such that :

- Each tour starts from and comes back to the depot
- Each customer appears exactly once in the set of tours
- The sum of the quantities ordered by the customer on any tour $\leq Q$
- + eventually other constraints on the tour length, time windows, multiple depots, etc.

Objective :

Minimize the total length performed by the vehicle

POPMUSIC FOR THE VEHICLE ROUTING PROBLEM



A solution can be decomposed into somewhat independent parts

A subset of part (sub-problem) can be optimized almost independently

Location-routing problem

Given :

n customers

m potential depots locations

q_i : quantity ordered by customer i

Travel costs between each pair of customers and between depots and customers

D : Depot opening cost

Q : vehicle capacity

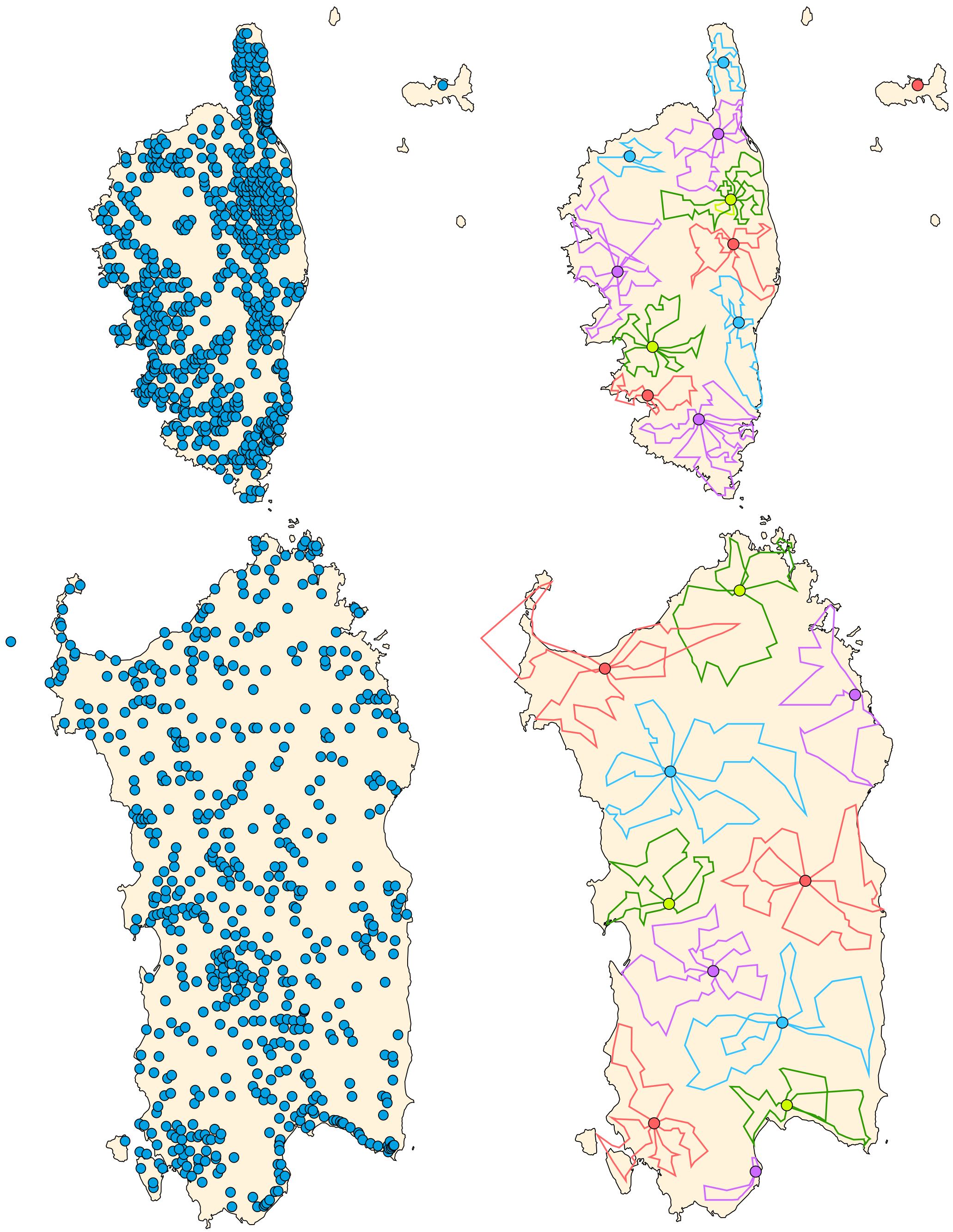
Find :

Subset of depots to open

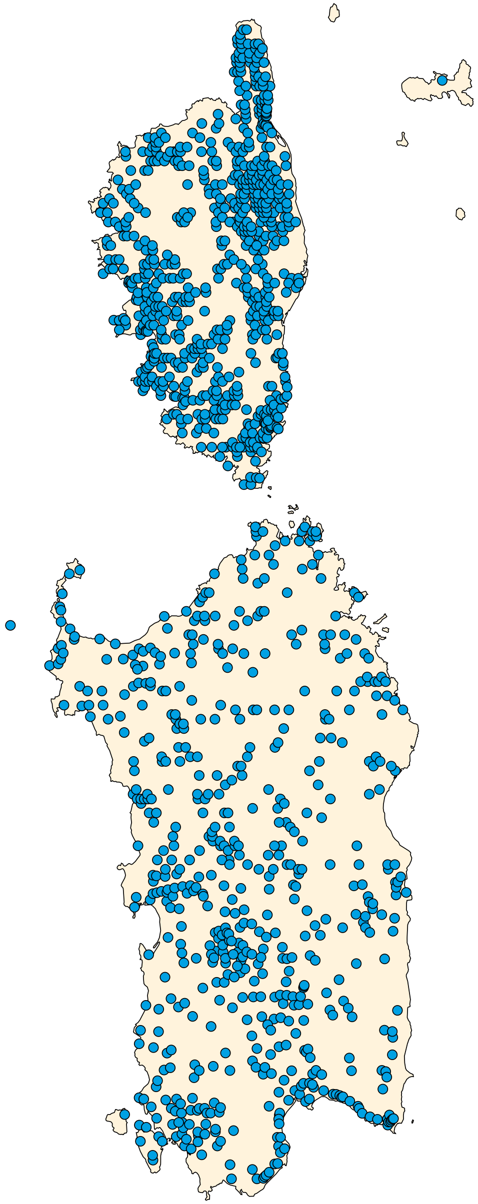
Set of tours verifying VRP constraints

Objective :

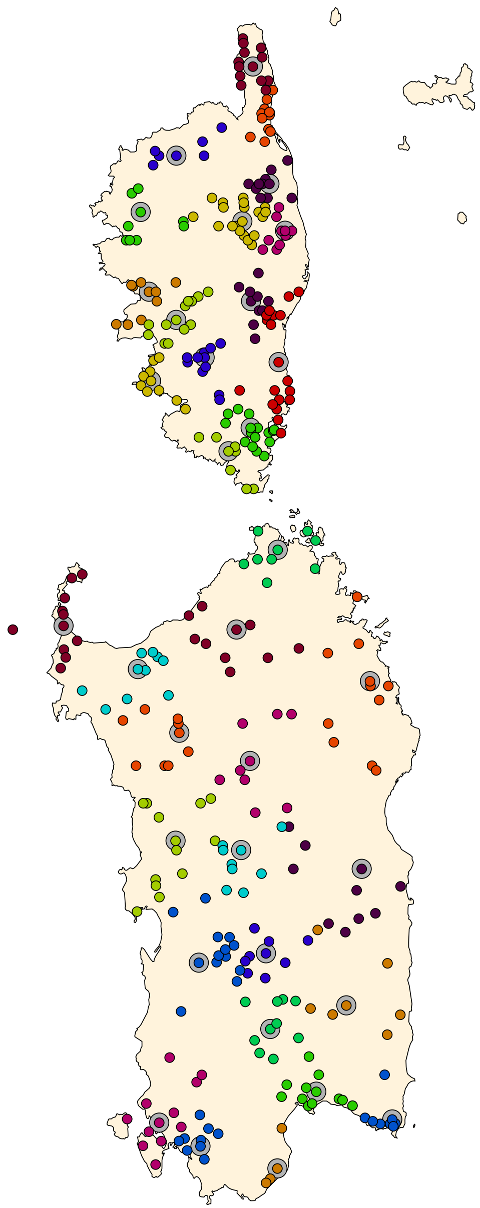
Minimize the total costs



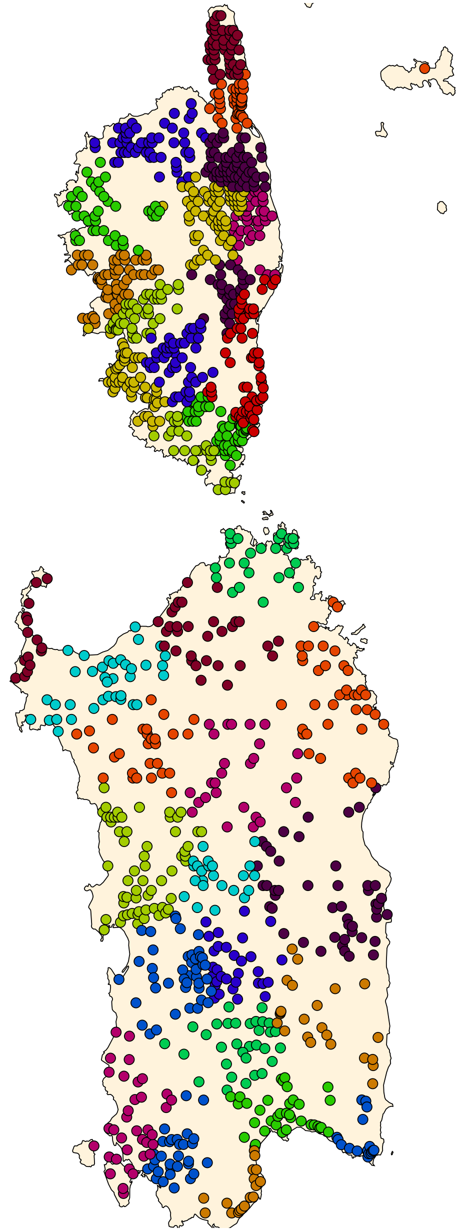
PROBLEM DECOMPOSITION



Initial set of elements



Sample + clustering

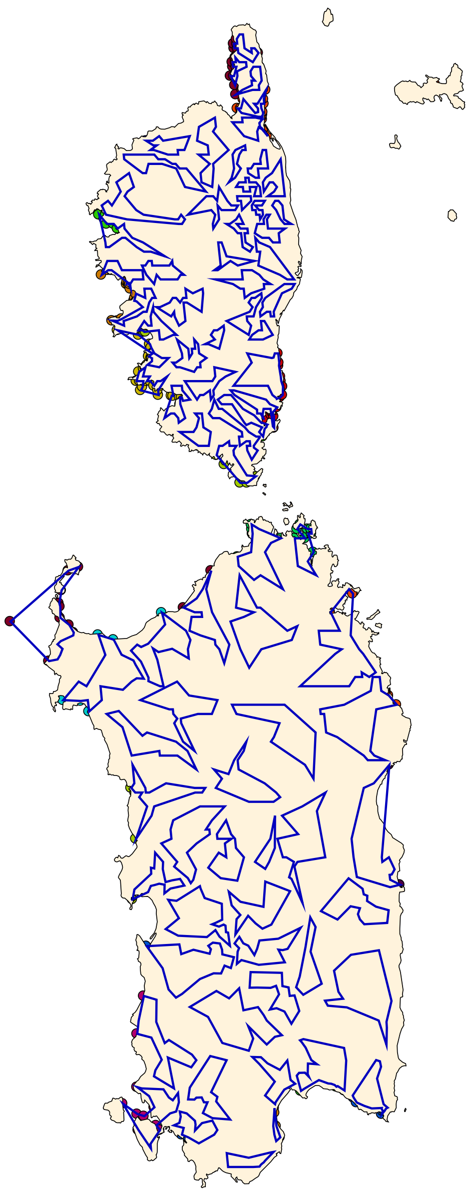


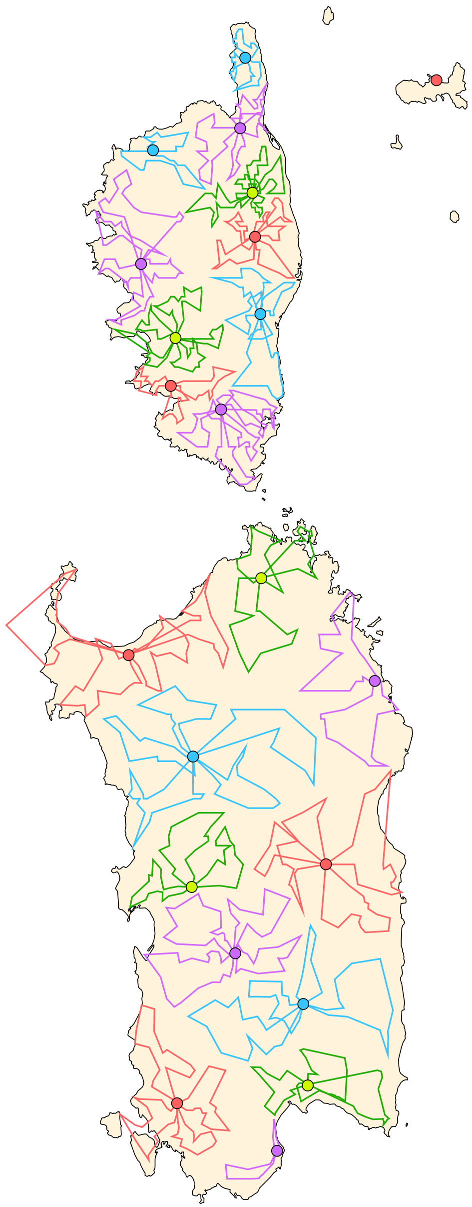
Clustering on all entities

APPLICATION TO LOCATION-ROUTING

Decomposition of clusters into smaller clusters that satisfy +/– vehicle capacity

Building independent vehicle tours

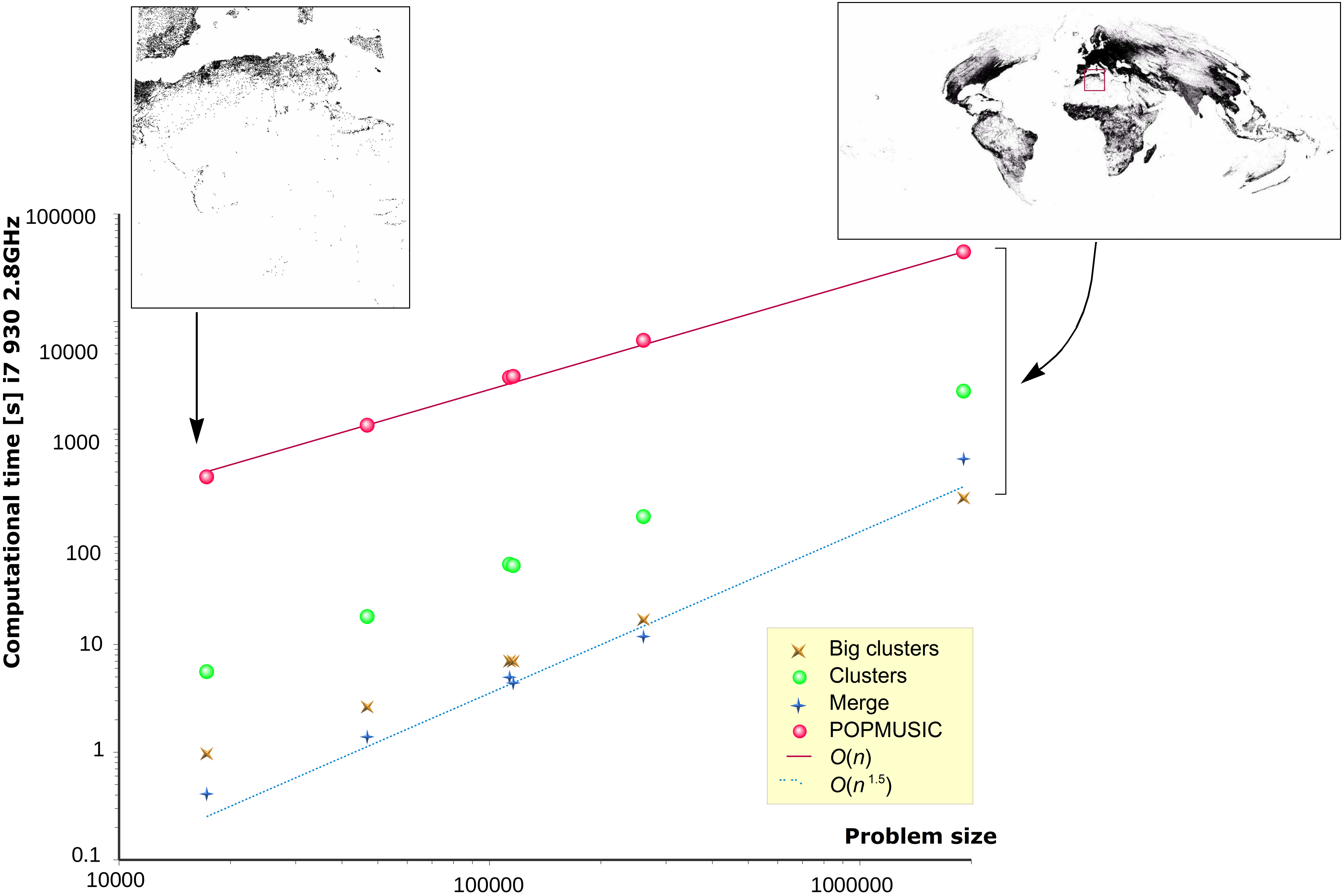




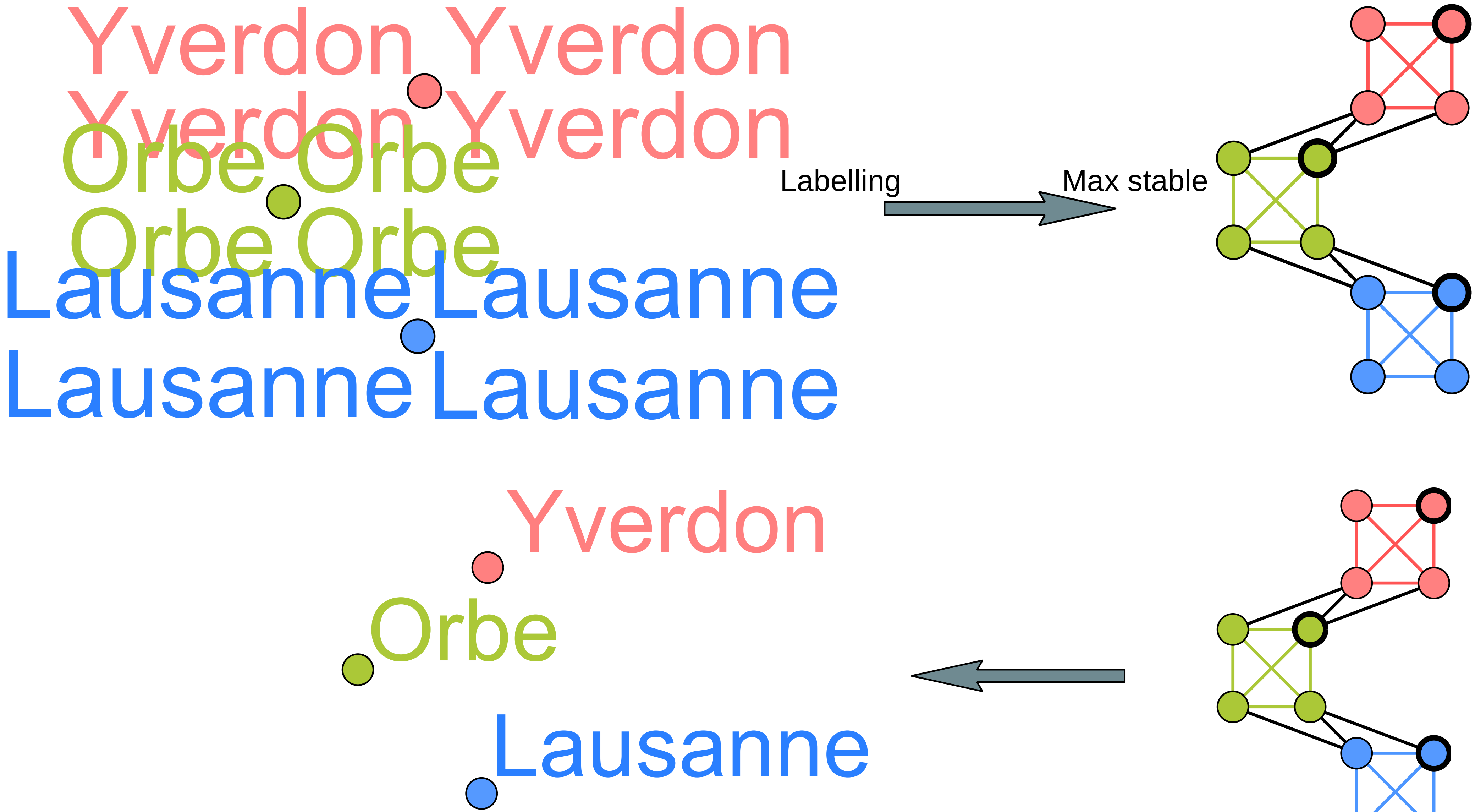
Finding depot location

Connection of TSP tours on depots

LOCATION-ROUTING : EMPIRICAL ALGORITHMIC COMPLEXITY



MAP LABELLING



Other problem that can be modelled like this : assigning flight levels and departure times of aircraft

POPMUSIC CHOICES FOR MAP LABELLING

Part:

Object to label

Distance between parts:

Minimum number of edges needed to connect parts

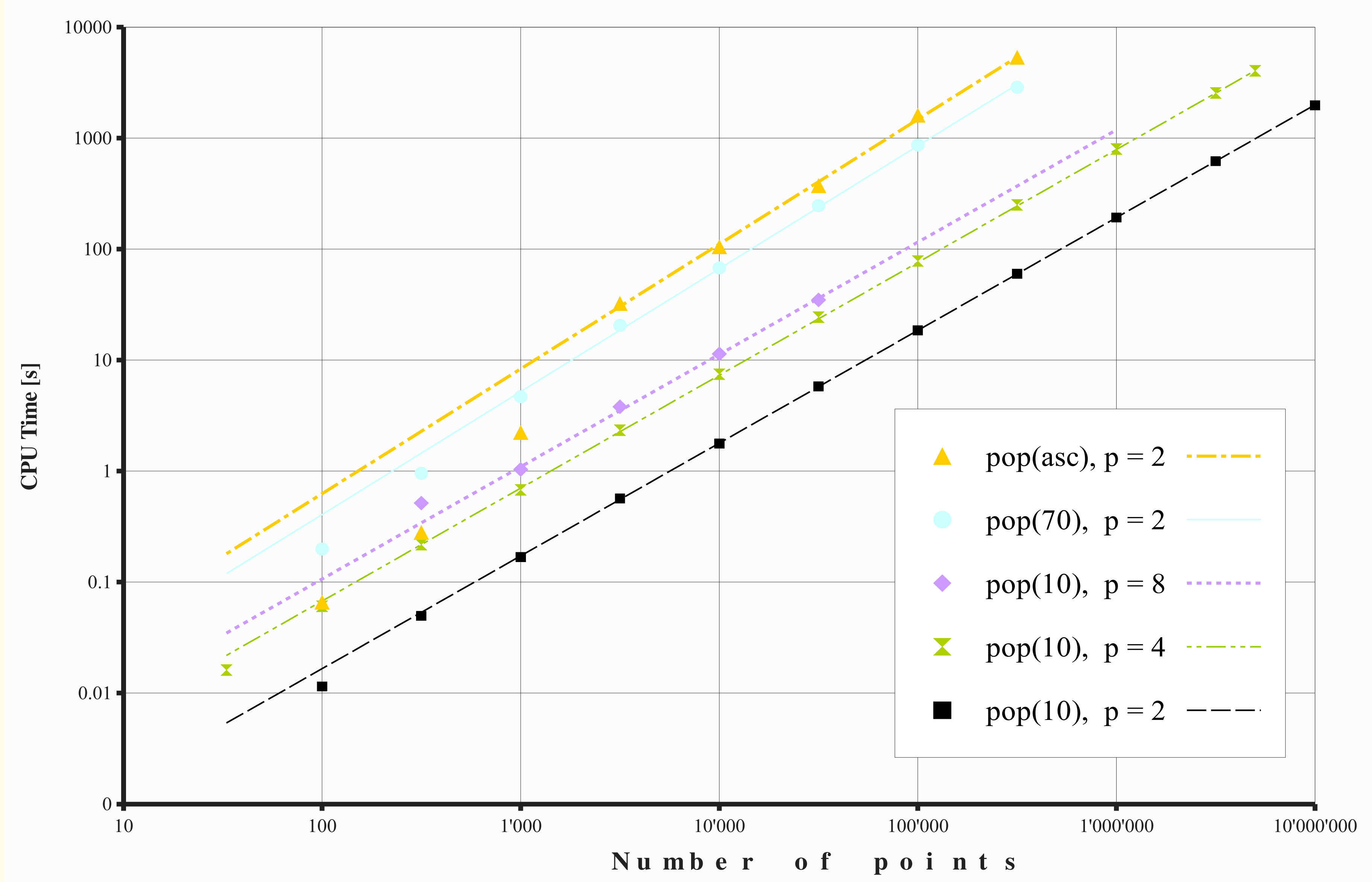
Vertex ≡ object

Edge ∃ possible conflict in labelling the objects associated to vertices connected

Optimization process:

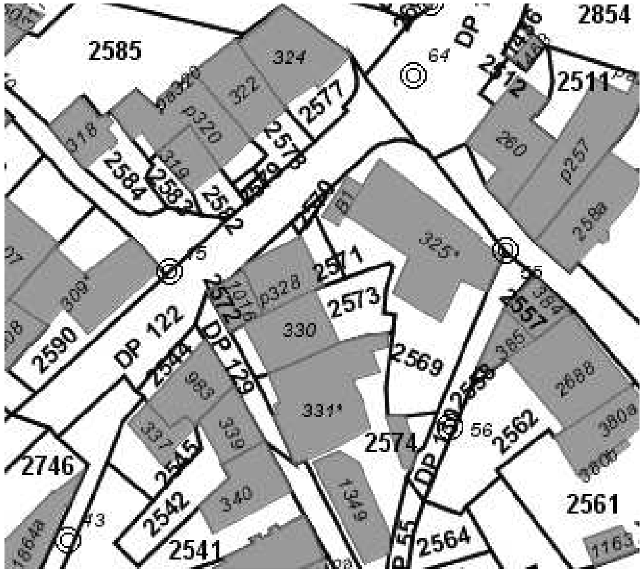
Tuned taboo search (Yamamoto, Camara, Nogueira Lorena, 2002), local search with ejection chains

Uniformly generated problem instances, between 30% and 90% of labels without overlap

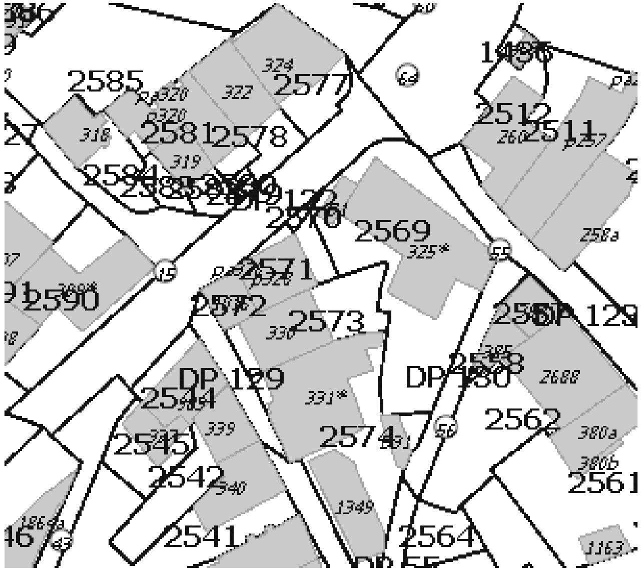


The complexity grows typically quasi-linearly with problem size

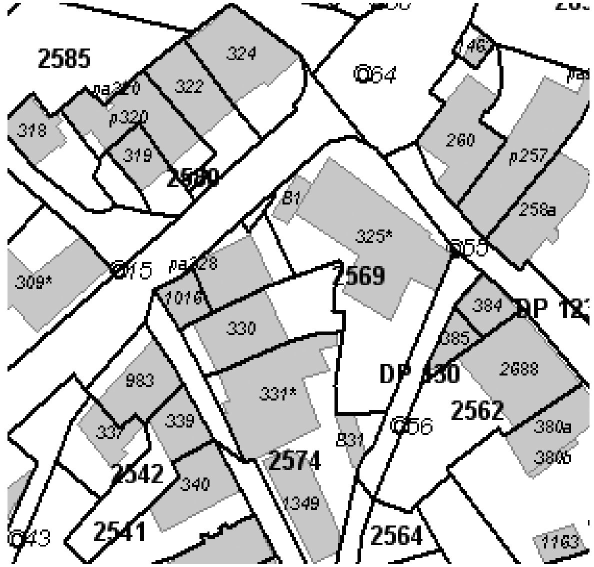
SUBJECTIVE RESULTS



POPMUSIC

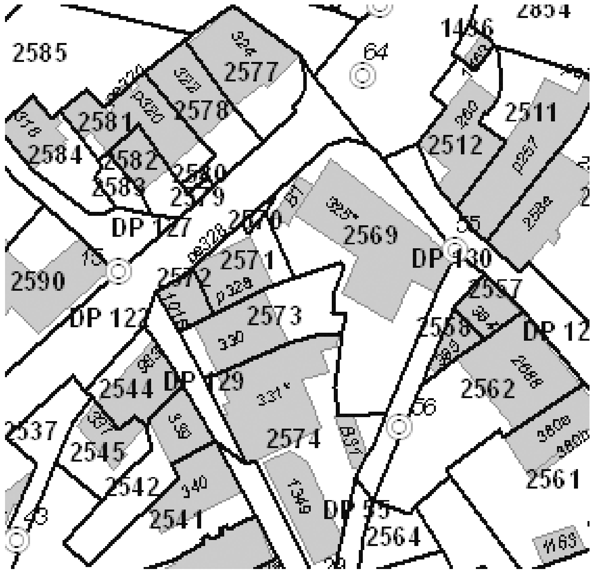


Quantum GIS (before)



Mapinfo professional

ArcGIS



ArcGIS + Maplex

Conclusions

POPMUSIC complexity

Can be implemented in $O(n^{3/2})$

Main difficulty : generating an initial solution, finding the *r* closest parts

POPMUSIC options and parameter

Natural stopping criterion

Must have an optimization process for sub-problems

- Heuristic
- Exact $\Rightarrow$ Matheuristic

A single parameter *r*, for defining sub-problem size

$\Rightarrow$ Easy to tune : sub-problem size must meet best efficiency of optimization process

POPMUSIC drawback

Definition of part and sub-problem dependent on problem under consideration

Application to higher dimensional instances

Up to now : Map labelling 2D, Location-routing $2^{1/2}$D, MDVRPTW 3D

What happens for higher dimensions ?

Application to other problems

Testing different definitions for parts

Study of different options

Definition of distance between parts

Management of non-optimized parts

Parallel implementations

LARGE NEIGHBOURHOOD SEARCH (LNS)

Idea

In an enumeration method for integer or mixed integer linear programming

- Fix the value of a subset (a majority) of variables
- Solve optimally the sub-problem on the remaining variables
- Repeat with other subsets of fixed variables

Evolution

Destroy a portion (free variables) of the solution

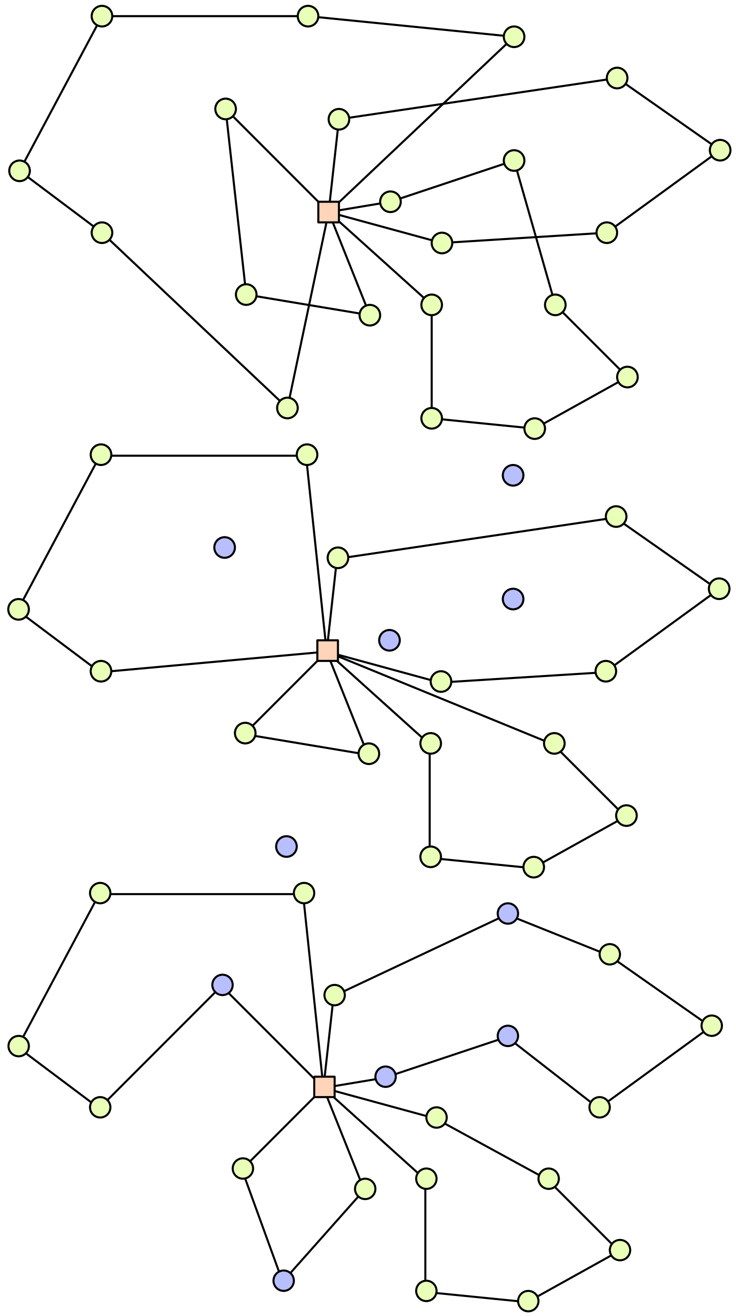
Try to rebuild the solution by keeping fixed variables

Repeat with other portions

Iterated local search

- Randomly perturb the best solution known
- Apply an improving method with penalties
- Repeat after having modified the penalties

LNS FOR THE VRP (SHAW 1998)



Generate an initial solution

Destroy mechanism

Select a random customer

and few close customers

"close" : Euclidean distance + random component

Repair method

Optimal or heuristic re-insertion (with constraint programming)

⇒ Applied to small-medium problem instances only

⇒ No preoccupation on algorithmic complexity

⇒ Destroy + repair = reoptimize a portion of the solution

POPMUSIC GENERAL IDEA

Start from an initial solution

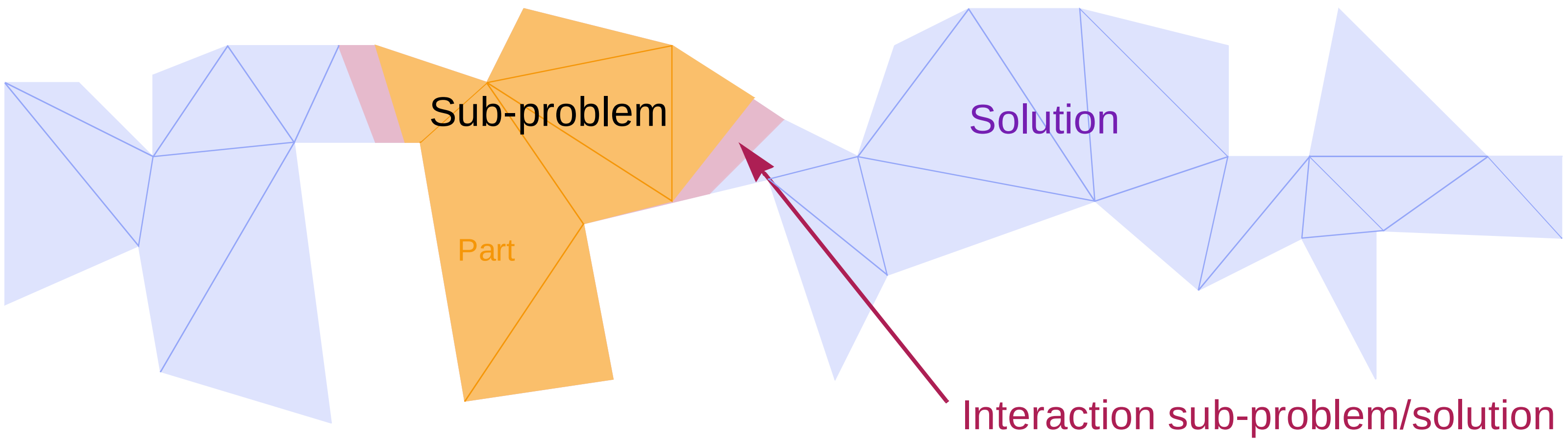
Decompose solution into parts

Optimize a portion (several parts) of the solution

Repeat, until the optimized portions cover the entire solution

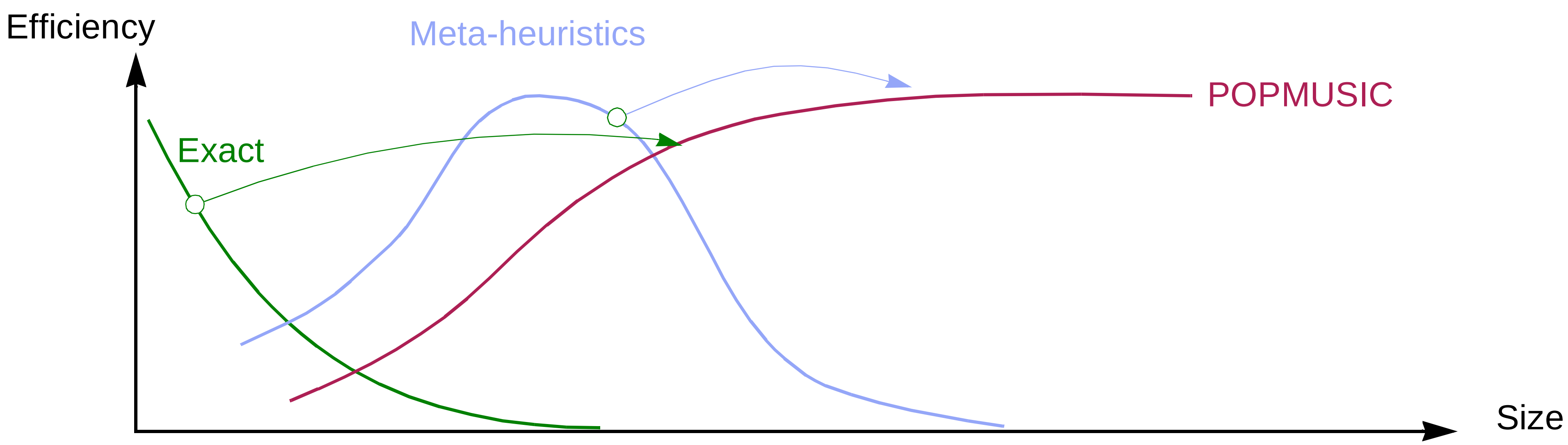
Difficulty

Sub-problems are not necessarily completely independent one another



CLASSIFICATION OF PROBLEM SIZE

Class	Typical technique	Size (order)	
Toy	Complete enumeration	10^1	
Small	Exact method	10^1—10^2	
Medium	Meta-heuristics	10^2—10^4	Memory limit $O(n^2)$
Large	Decomposition techniques	10^3—10^7	Time limit $O(n^{3/2})$
Very Large	Distributed database	above	



Input

n elements, function $d(i, j)$ measuring the proximity between elements i and j

Body

1 Create a random sample E of $20\sqrt{n}$ elements

2 Solve a relaxation of a p-median with capacity with $p = \sqrt{n}$ on E

3 Assign each of the n elements to its closest among the p centres
$\Rightarrow$ $\sqrt{n}$ clusters with $\sim\sqrt{n}$ elements each

4 Build a proximity graph G on the centres
$\Rightarrow$ c_i and c_j are neighbours if:
there is an element assigned to c_i which second closest centre is c_j

Output

$\sim\sqrt{n}$ clusters, proximity graph G

POPMUSIC FOR LOCATION-ROUTING (ALVIM & TAILLARD 2012)

Part:

Vehicle tour

Distance between parts:

Minimal distance between customers of different tours

A sub-problem is a smaller MDVRP

Optimization process:

Basic tabu search for MDVRP

Particularity:

No depot relocation