Chapter 4

Decomposition Methods

Size of problem instances

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	
Very Large	Distributed database	above	

Large neighbourhood search

Popmusic : a generic decomposition technique

Applications

VRP

Clustering

Cartographic labelling

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

POPMUSIC GENERAL IDEA

Start from an initial solution

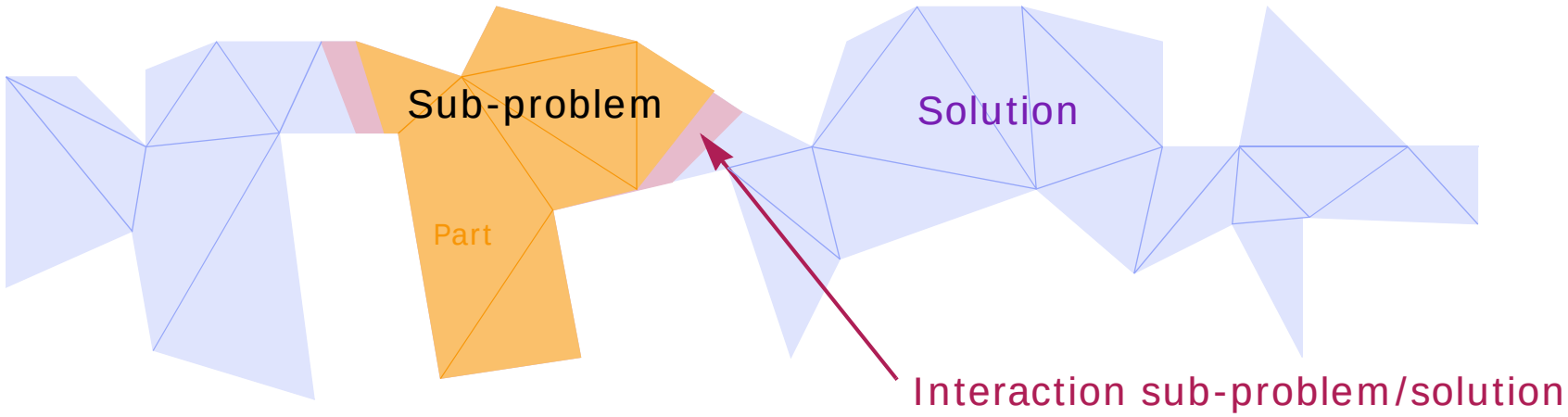
Decompose solution into parts

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



TEMPLATE 4.1 : POPMUSIC

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 $s_i \notin O$
   // r : parameter
2. Create a sub-problem R composed of the r " closest " parts $\in S$ from s_i
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

Definition of a part

Distance between two parts

Parameter *r*

Optimization procedure

Variants :

Slower :

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

Faster :

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

RELATED CONCEPTS

Candidate list, strongly determined and consistent variables (Glover)

“Chunking” (Woodruff)

Large neighbourhoods (Shaw)

VDNS (Hansen & Mladenovic)

Decomposition methods

POPMUSIC FOR VRP (TAILLARD 1993, ...)

Part:

Vehicle tour

Distance between parts:

Polar distance between centres of gravity

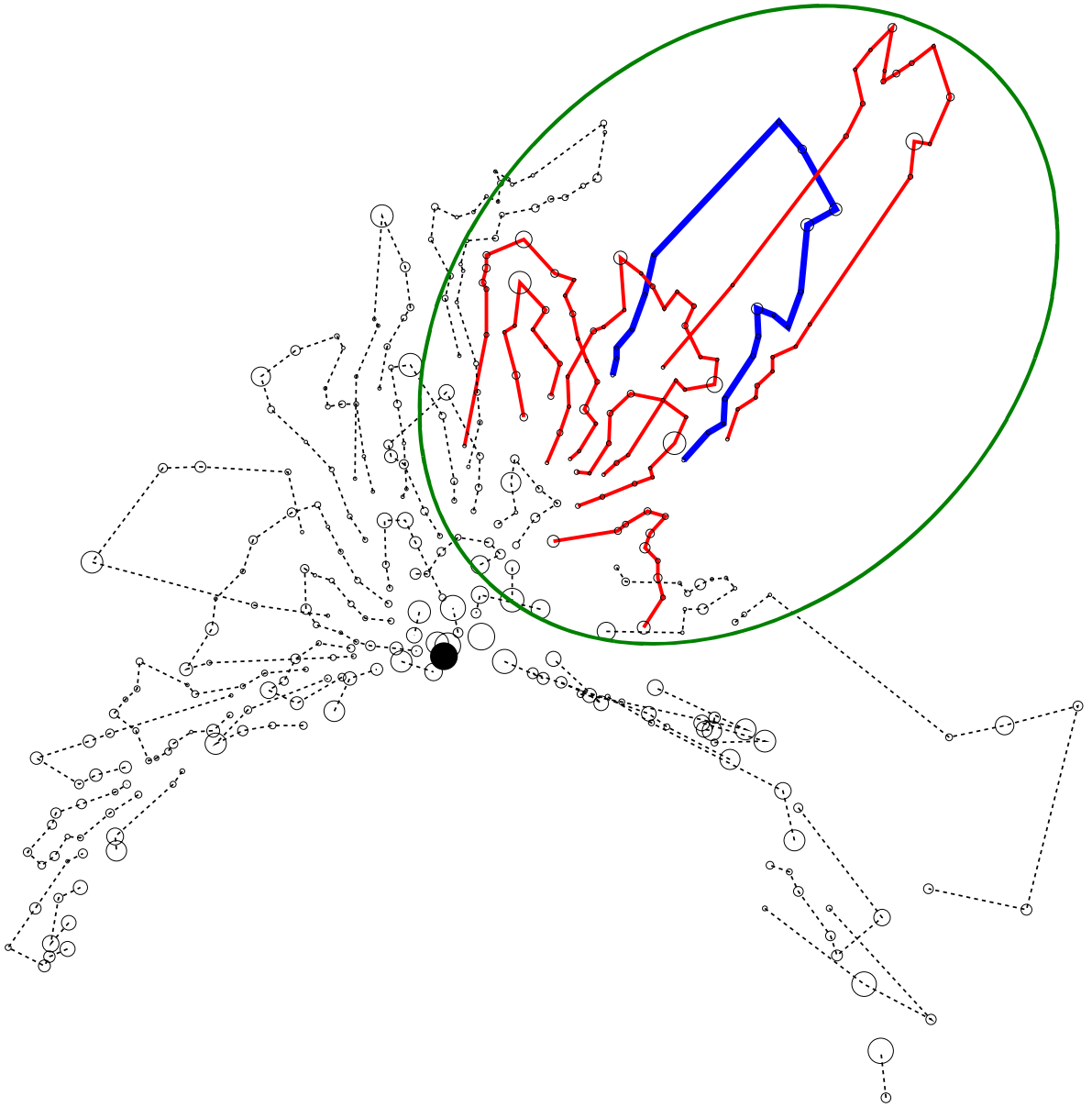
A sub-problem is a smaller VRP

Optimization process:

Basic tabu search

Particularity:

Many simultaneous optimization processes, treating all tours at each iterations



LNS FOR THE VRP (SHAW 1998)

Part

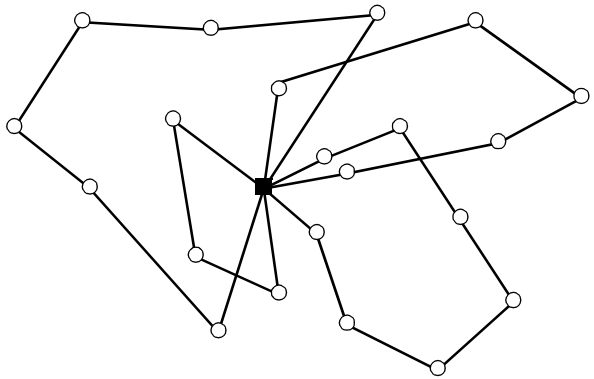
Individual customers

Distance between parts

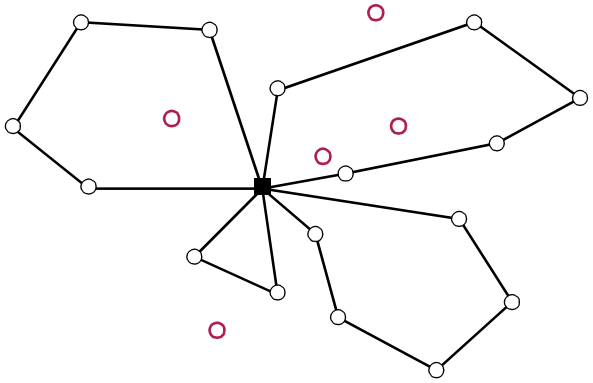
Euclidean distance + random component

Optimization process

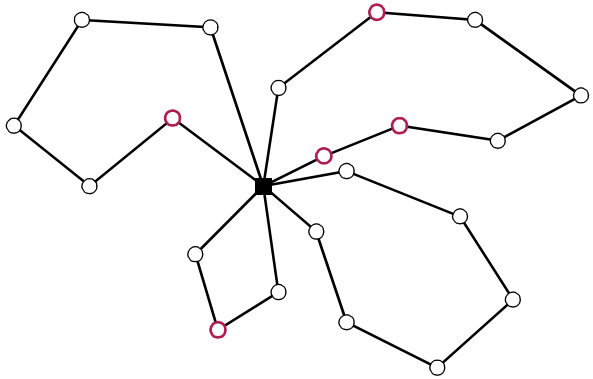
Optimal or heuristic re-insertion (with constraint programming)



Initial solution



Customers removal



Optimal re-insertion

POPMUSIC FOR CLUSTERING

Part :

Elements belonging to a cluster

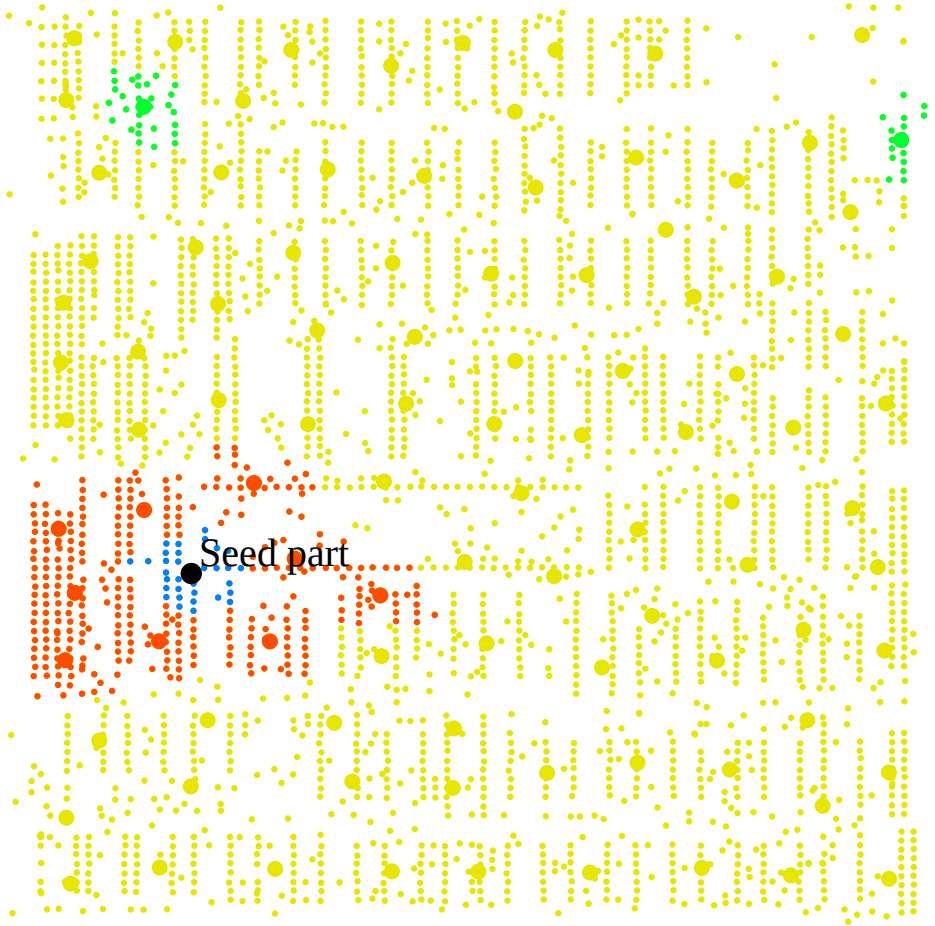
Distance :

Average dissimilarity between elements of different groups,

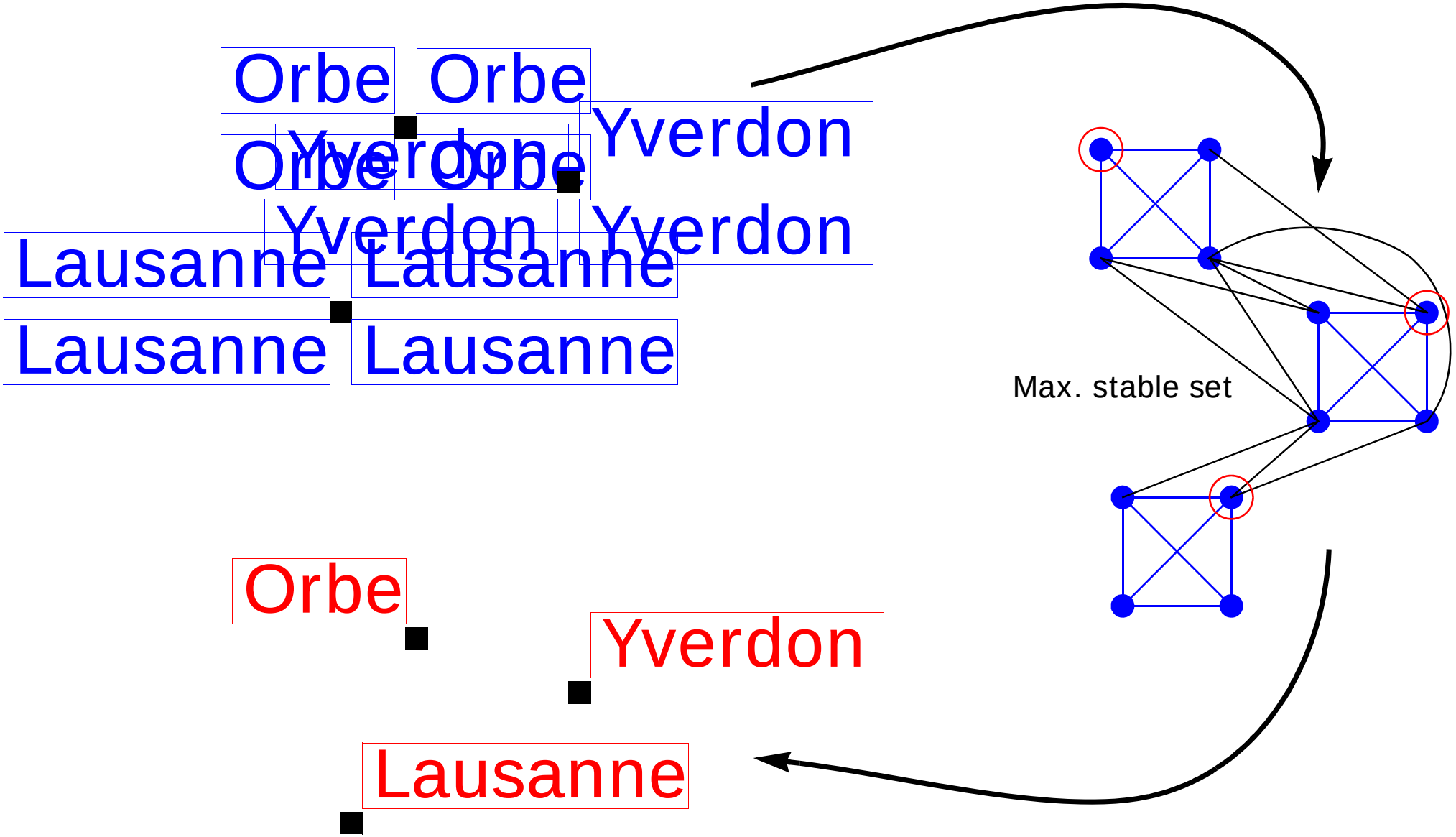
Distance between centres

Optimization process :

Improving method based on candidate list, relocation of a centre, stabilizing solutions (CLS)



CARTOGRAPHIC LABEL PLACEMENT



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

POPMUSIC CHOICES

Part:

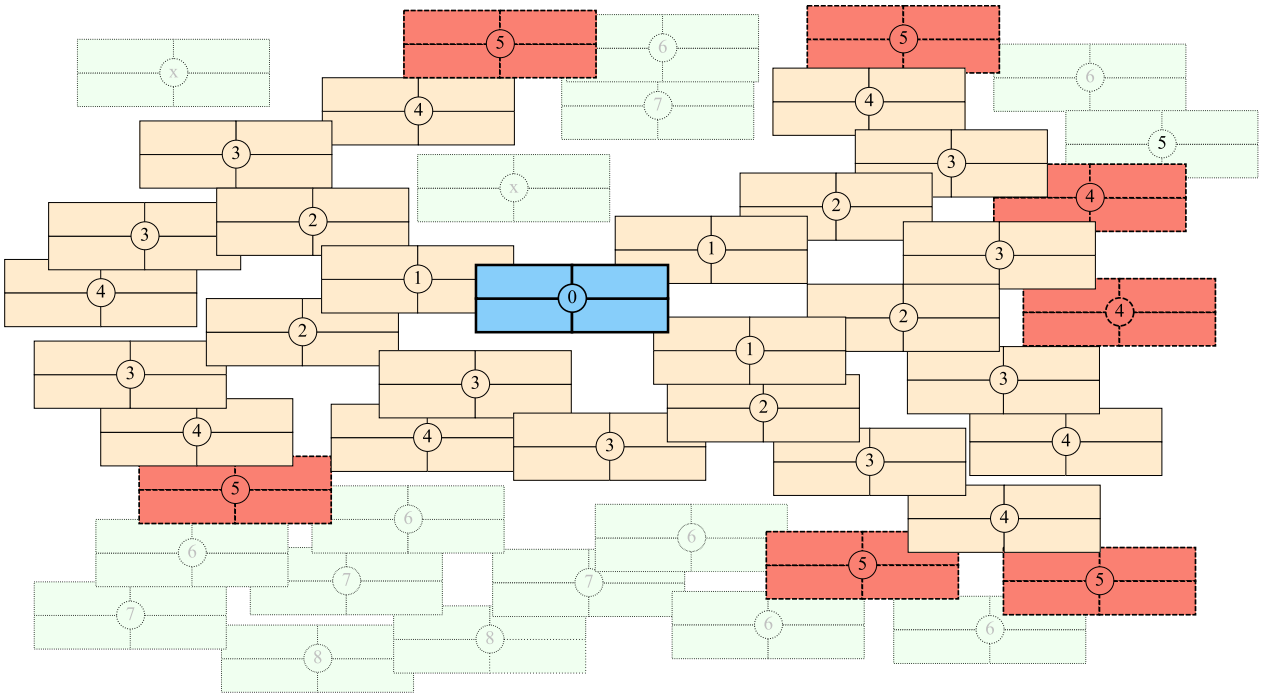
Object to label

Distance between parts:

Minimum number of edges needed to connect parts

Vertex ≡ object

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

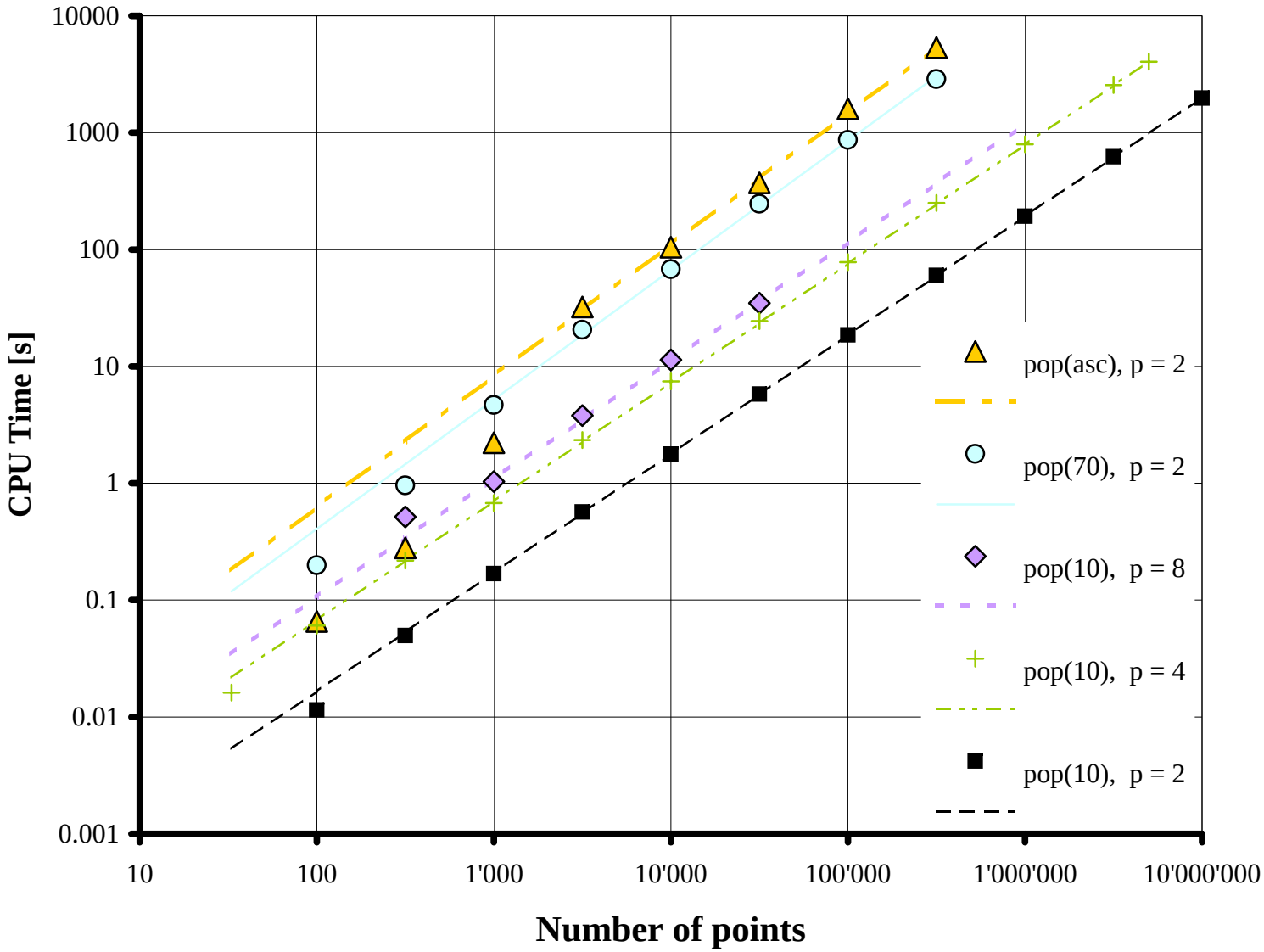


Optimization process:

Tuned taboo search (Yamamoto, Camara, Nogueira Lorena, 2002)

NUMERICAL RESULTS

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



The complexity grows typically quasi-linearly with problem size

EXERCISES

Exercise 4.1

In the context of POPMUSIC, how to define a part and a sub-problem for the permutation flow shop problem ? How the interaction between the sub-problem and the other parts must be taken into account ?

Exercise 4.2

If the size of the sub-problems is independent of the size of the problem, then it can be considered that a sub-problem can be solved in constant time. Empirical observations, such as those presented on the figure on Page 106 shows that a part is put in set *O* a number of times that seems to be almost independent from the problem size, on the average. If these hypotheses are satisfied, what are the most complex part of POPMUSIC template (in term of algorithmic complexity) ?