

Ant Colonization Optimization : Comparison of Elitist Ant and Rank-Based Ant on the Quadratic Assignment Problem

Daoud Yemni¹

¹Université Libre de Bruxelles, Brussels
dyemni@ulb.ac.be

June 3, 2018

Abstract

This article will talk about one of the hardest problems in NP-hard class. The *Quadratic Assignment Problem (QAP)* is a problem in the branch of optimization and operations research from the category of facilities location problems. To solve the problem, we will implement an Ant Colonization Optimization system and compare their results and performance. The algorithms used are *Elitist Ant System* and *Rank-Based Ant System*. The procedure consists in implementing both algorithm, simulate them and compare their results. Then we will take one of algorithm, add the local search and compare the results. At the end, there is a discussion about which one is recommended for solving the *QAP*.

1 Introduction

The *Quadratic Assignment Problem* is one of the hardest problem in NP-hard class. One of applications of QAP is to assign a set of facilities to a set of locations while minimizing the total assignment cost. We formulate the QAP as said before. We define a set of facilities and a set of locations with given distances between locations and given flows between facilities. The goal will be to minimize the total cost of distances and flows. Given n facilities, n locations, two $n \times n$ matrix, one matrix of distances with given distances $A[a_{ij}]$ and one matrix of flows

$B[b_{ij}]$, the objective function of this problem is formulated as :

$$\psi \in S(n) = \sum_{i=1}^n \sum_{j=1}^n b_{ij} \cdot a_{\psi_i \psi_j}$$

where $S(n)$ is the set of all permutations from the set $\{1, \dots, n\}$ and ψ_i gives the location of facility i in the current solution. For solving the problem, we will define the Ant Colonization System and his formulas. We will explain and implement two specific ant system. The two specific algorithms will be : *Elitist Ant System* and *Rank-Based Ant*. A section will discuss about the *Automatic Tuning Configuration* or *Self-tuning*. This is a technique for finding the best configuration of an algorithm for a given set of configurations and a set of instances. The problem will be simulated and solved by both algorithms. The procedure of solving is each ant will produce his own assignment by visiting the n locations. Then the experiments will be explained, simulated and the results will be reported. Finally, a discussion about which one of the algorithm is recommended to solve the problem.

2 Ant Colonization System

The Ant Colonization System consists in producing a set of ants and compute the solution of the objective function. Each ant will provide an evaluation of the objective function. The

construction of the evaluation is simple. Each ant will create his assignment/path in which corresponds to the visited cities in a certain order. The path of the ant depends on the heuristic and the pheromone trails between locations. The heuristic for the problem is defined as :

$$h_{i,j} = \frac{1}{a[i,j] \cdot b[i,j] + 1}$$

where a is the distance matrix and b is the flows matrix.

The pheromone trails are updated as follows :

$$\begin{cases} \tau_{ij}(0) = \frac{1}{\rho \cdot N} \\ \tau_{ij}(t) \leftarrow (1 - \rho) \cdot \tau_{ij}(t-1) + \Delta\tau_{ij} \end{cases}$$

where $\Delta\tau_{ij} = \sum_{k=1}^m \Delta\tau_{ij}^k$, $\Delta\tau_{ij}^k = \frac{1}{L^k}$ in which L^k is the length of the solution found by the ant k and N is the number of city.

The computation of the probability is still the same that the basic from TSP :

$$P_{i,j} = \frac{ph[i,j]^\alpha \cdot h[i,j]^\beta}{\sum_{q \in J} ph[i,q]^\alpha \cdot h[i,q]^\beta}$$

where ph is the pheromone matrix, the denominator represents the memory of the ant.

I decide to compare the Elitist Ant system and the Rank-Based Ant system. They use the same ethics that the best ant(s) can add more pheromone in the pheromone trails. Therefore, I implemented those algorithms to show which one give a better a solution for the problem of QAP.

2.1 Elitist Ant System

The Elitist Ant System is an extension of the ACO. The ant(s), that have the best solution, can add a plus-value in the pheromone trails. In this algorithm, the updating function of the pheromone trails is a little bit modified. We add to the solution the number of best times the delta of the best ant(s) as follow. Then, the update of pheromone trails is defined as :

$$\tau_{ij}(t) = (1 - \rho) \cdot \tau_{ij}(t-1) + \sum_{k=1}^m \Delta\tau_{ij}^k + e \cdot \Delta\tau_{ij}^{bs}$$

where bs means the best solution and $\Delta\tau_{ij}^{bs} = \begin{cases} \frac{1}{L^{bs}} & if arc(i,j) \in T^{bs} \\ 0 & otherwise \end{cases}$

2.2 Rank-Based Ant System

The Rank-Based Ant is an extension of the ACO. The goal of the Rank-Based Ant algorithm is the ants are ranked regarding their solution. Every ants can add pheromone and the impact of their pheromone depends on their rank. More the ant is ranked, more the impact of his pheromone is big :

$$\tau_{ij}(t) = (1 - \rho) \cdot \tau_{ij}(t-1) + \sum_{r=1}^{\omega-1} (\omega - r) \Delta\tau_{ij}^r + \omega \cdot \Delta\tau_{ij}^{bs}$$

where ω is the number of top-rank ants allowed to contribute in the pheromone trails.

3 Automatic Tuning Configuration

Automatic Tuning Configuration is a technique for finding the best configuration of an algorithm by given a set of configuration and a set of instances. There exists such ways to find the best configurations. You can find the best configuration by implementing a Machine Learning technique or by using a statistical function to evaluate the results of a configuration. The technique used in my project was inspired about the ParamILS. Given a set of configuration $\Theta = \{\theta_1, \theta_2, \dots, \theta_k\}$ and a set of instances $X = \{X_1, X_2, \dots, X_l\}$. I will check each configuration $\theta_i \in \Theta$ on each instance $X_j \in X$ and retrieve the best configuration for the instance θ_j . At the end, we remain the configuration that satisfied the most instances, θ_j^{best} . Then the remaining configuration θ_j^{best} is saved in a file (so one file per algorithm) and is used for the main program.

My list of predefined configuration contains all of possible combinations (except the case where $\alpha = \beta = 0$) with those values :

- Number of ants : 5, 10 or 15

- α : 0 or 1
- β : 0 or 1
- ρ : 0.5
- ω : 4

Remark: The technique implemented is not the best one. We can find a different best configuration each time. But after several runs, some of configurations are often returned.

4 EAS vs RBA

In this section, we will compare the results obtained by both algorithm for each instances. Each algorithms are run 10 times on 10 different seeds for each instances. The results of the best solution for each seeds and each independent runs are stored and we compute the mean of the runs. The different seeds used for the experiments are :

1. $m = 5$, $\alpha = 0$ and $beta = 1$
2. $m = 5$, $\alpha = 1$ and $beta = 0$
3. $m = 5$, $\alpha = 1$ and $beta = 1$
4. $m = 10$, $\alpha = 0$ and $beta = 1$
5. $m = 10$, $\alpha = 1$ and $beta = 0$
6. $m = 10$, $\alpha = 1$ and $beta = 1$
7. $m = 15$, $\alpha = 0$ and $beta = 1$
8. $m = 15$, $\alpha = 1$ and $beta = 0$
9. $m = 15$, $\alpha = 1$ and $beta = 1$
10. $m = 20$, $\alpha = 1$ and $beta = 1$

The others parameters keep the default value : $\rho = 0.5$ and $\omega = 4$.

4.1 Results

4.1.1 Best solution quality

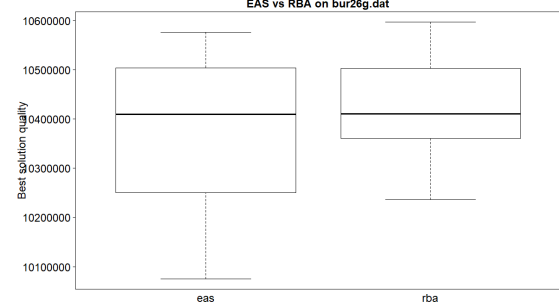


Figure 1: Results on the both algorithm on the instance bur26g

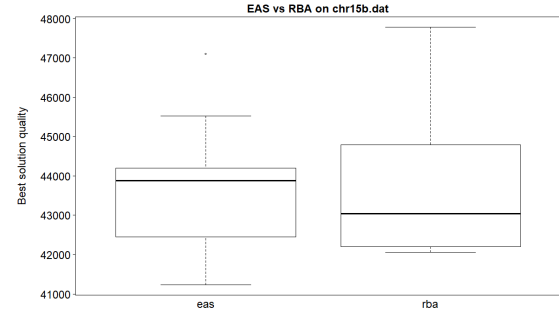


Figure 2: Results on the both algorithm on the instance chr15b

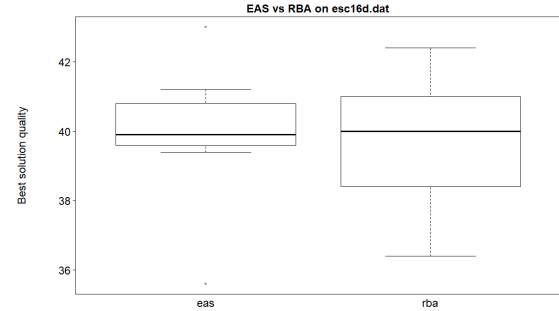


Figure 3: Results on the both algorithm on the instance esc16d

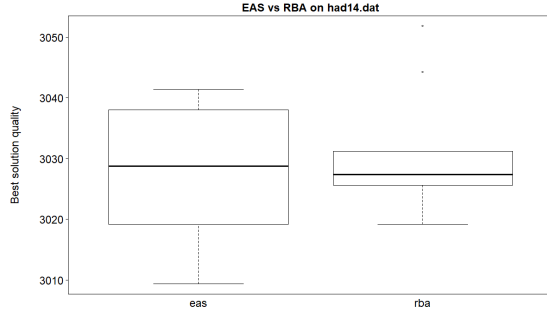


Figure 4: Results on the both algorithm on the instance had14

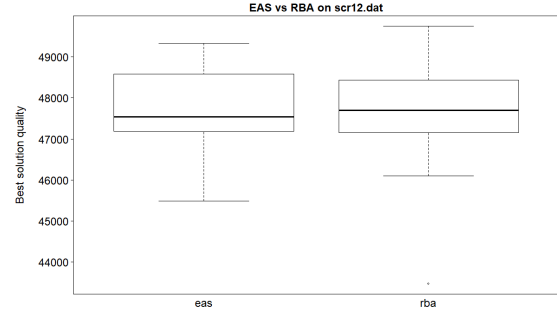


Figure 7: Results on the both algorithm on the instance scr12

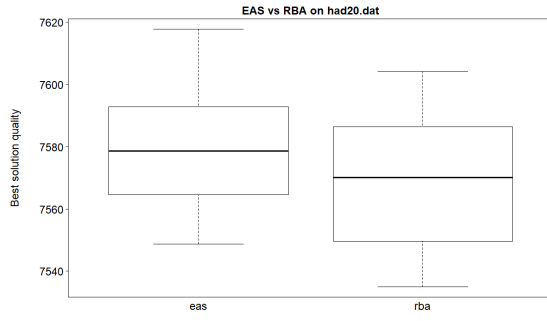


Figure 5: Results on the both algorithm on the instance had20

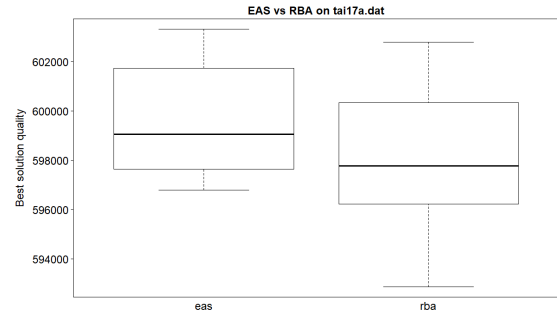


Figure 8: Results on the both algorithm on the instance tai17a

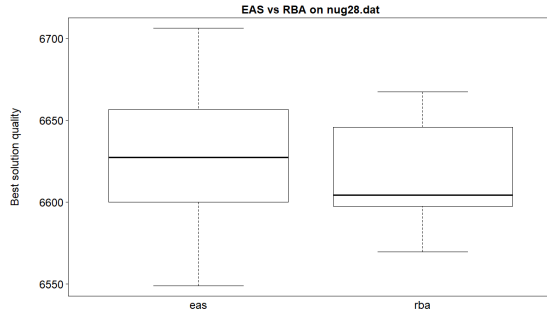


Figure 6: Results on the both algorithm on the instance nug28

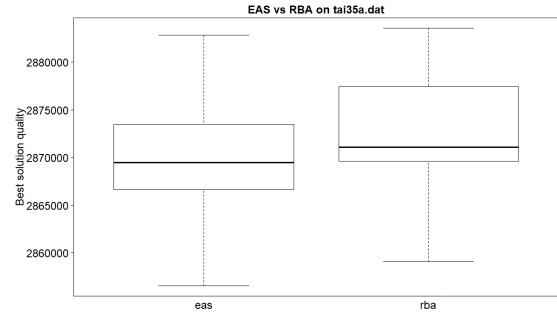


Figure 9: Results on the both algorithm on the instance tai35a

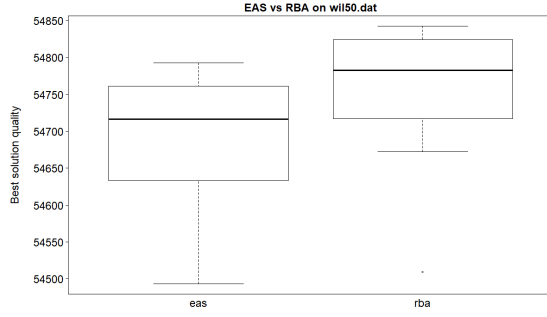


Figure 10: Results on the both algorithm on the instance wil50

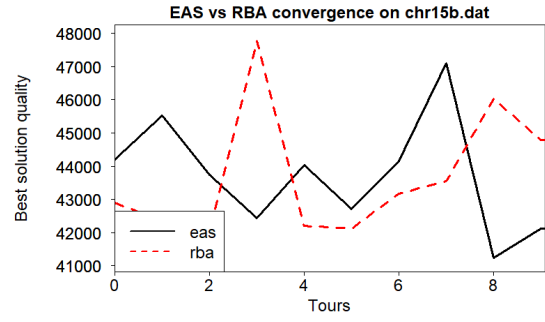


Figure 12: Results on the both algorithm on the instance chr15b

4.1.2 Wilcoxon

Wilcoxon signed rank (paired)

eas rba

eas NA 1

rba NA NA

Bonferroni correction

eas rba

eas NA 1

rba NA NA

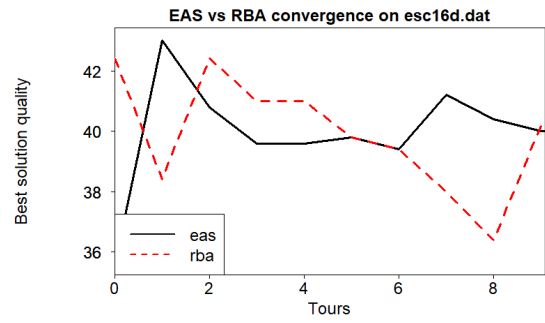


Figure 13: Results on the both algorithm on the instance esc16d

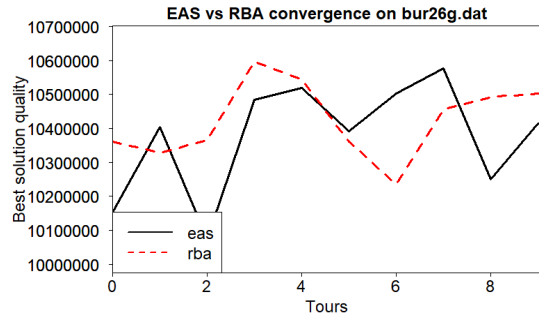


Figure 11: Results on the both algorithm on the instance bur26g

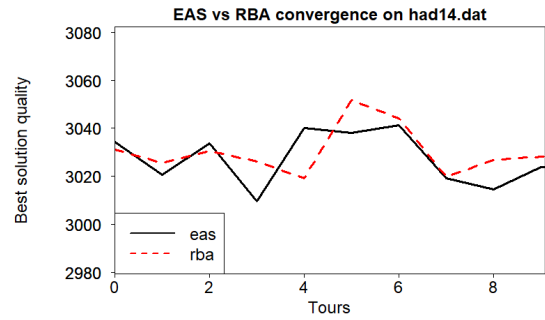


Figure 14: Results on the both algorithm on the instance had14

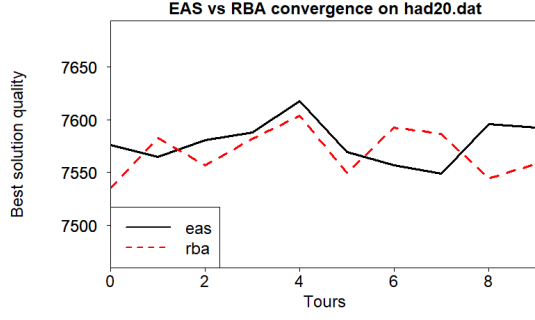


Figure 15: Results on the both algorithm on the instance had20

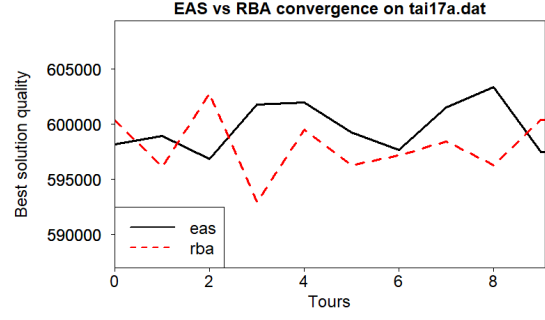


Figure 18: Results on the both algorithm on the instance tai17a

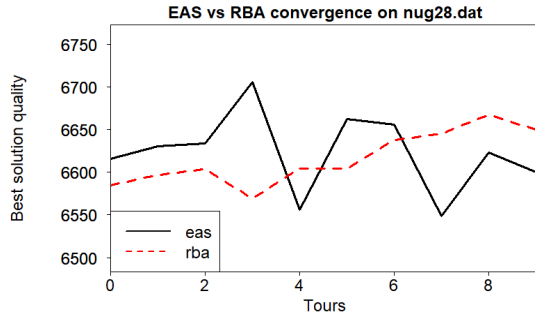


Figure 16: Results on the both algorithm on the instance nug28

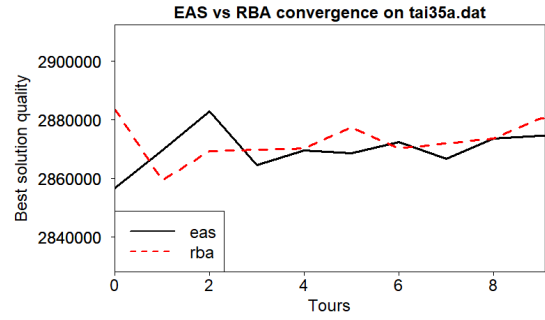


Figure 19: Results on the both algorithm on the instance tai35a

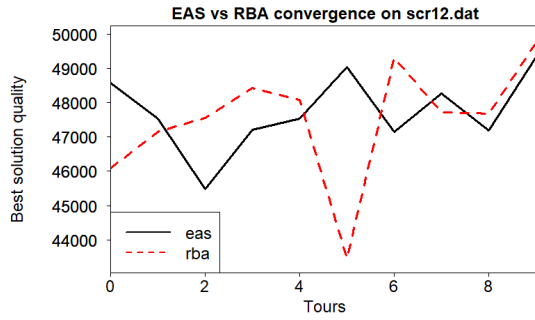


Figure 17: Results on the both algorithm on the instance scr12

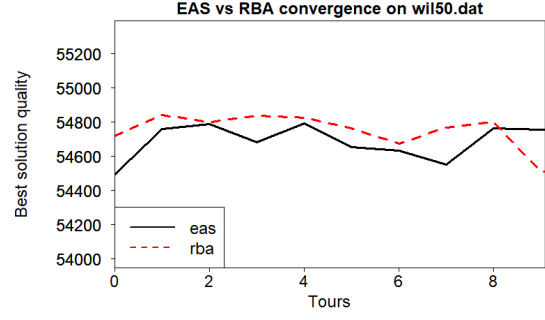


Figure 20: Results on the both algorithm on the instance wil50

4.2 Analysis

As we can see in the results, we can notice that the Elitist Ant system, in majority of instances, give better results than the Rank-Based Ant System. About the convergence, we can no-

tice that both algorithms are noisy in some instances but the behaviors are almost the same. In multiple experiments, I noticed that, sometimes, RBA can be better than EAS but in general, the EAS returns best results in most of cases. In majority, EAS returns best solutions but not necessarily a better performance. It means that the objective of only the best ant can add more pheromone is little bit better than the objective of adding pheromone depending on the rank.

So we can conclude that the **Elitist Ant system** is recommend for solving the Quadratic Assignment Problem.

5 EAS vs local search

In this section, we will compare the results and performance between the **EAS** and his local search version. We will reproduce the same experiments with the same set of seeds mentioned in the section 4. The local search applied, in this case, is the 2-opt algorithm. The 2-opt is algorithm in which we permute 2 edges and compare the results. If the results, after permutation, is better than before, the permutation is kept. In the case of Ant Colonization, When an ant finish to build his solution, we loop over the ant's path and permute 2 cities. If the permutation give a better cost, we save the permutation and we continue otherwise we revert the permutation until we reach the end of the path ($\Theta(N)$ where N is the number of city).

5.1 Results

5.1.1 Best solution quality

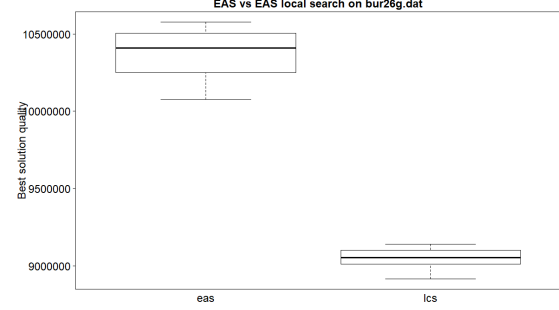


Figure 21: Results on the both algorithm on the instance bur26g

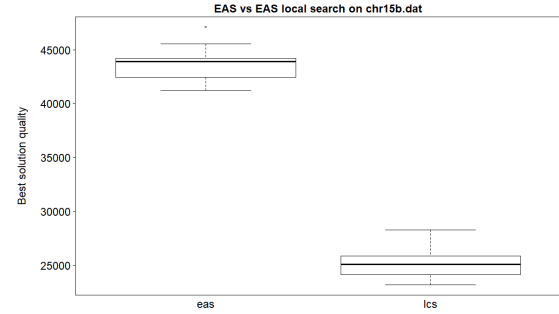


Figure 22: Results on the both algorithm on the instance chr15b

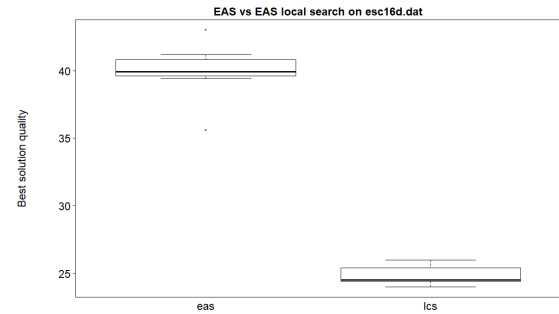


Figure 23: Results on the both algorithm on the instance esc16d

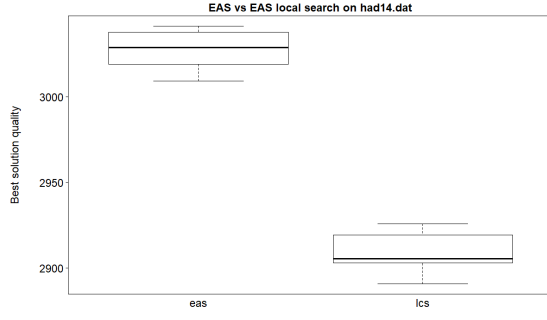


Figure 24: Results on the both algorithm on the instance had14

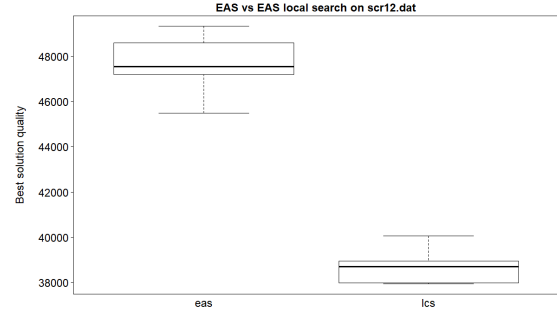


Figure 27: Results on the both algorithm on the instance scr12

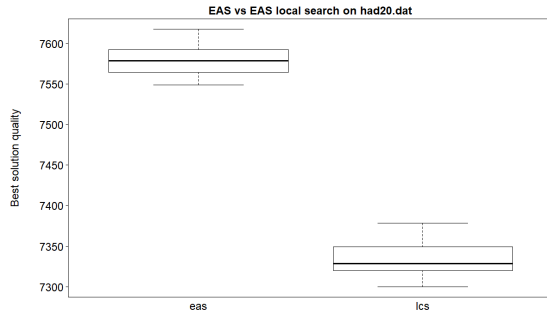


Figure 25: Results on the both algorithm on the instance had20

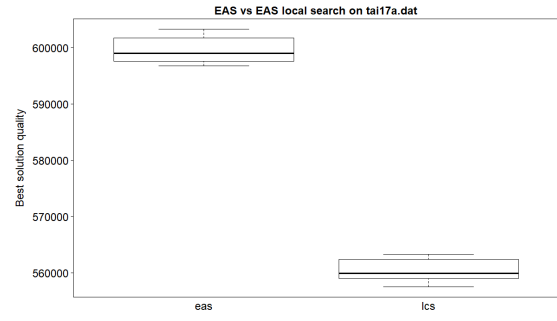


Figure 28: Results on the both algorithm on the instance tai17a

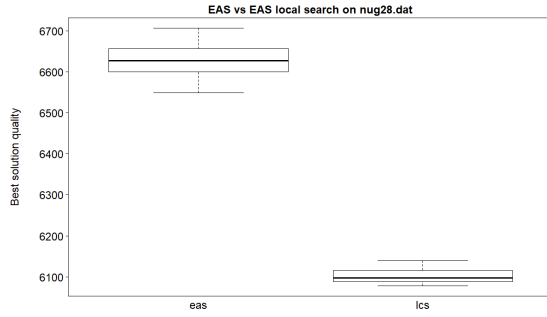


Figure 26: Results on the both algorithm on the instance nug28

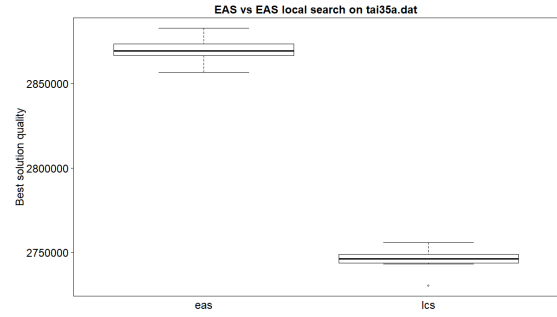


Figure 29: Results on the both algorithm on the instance tai35a

5.1.2 Wilcoxon test ranked

Wilcoxon signed rank (paired)

	eas	lcs
eas	NA	0.001953125
lcs	NA	NA

Bonferroni correction

	eas	lcs
eas	NA	0.001953125
lcs	NA	NA

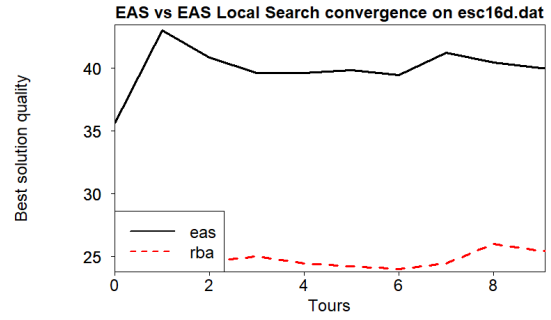


Figure 32: Results on the both algorithm on the instance esc16d

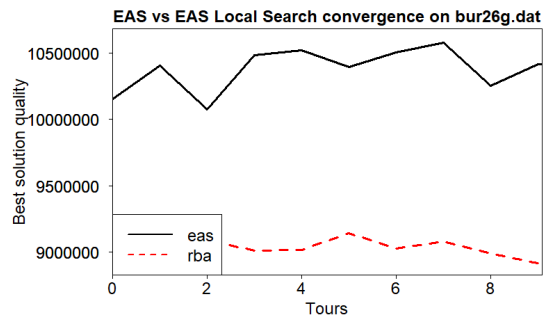


Figure 30: Results on the both algorithm on the instance bur26g

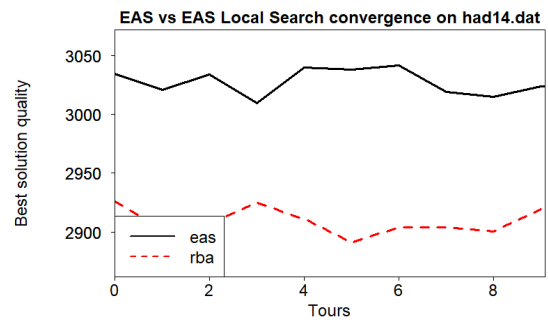


Figure 33: Results on the both algorithm on the instance had14

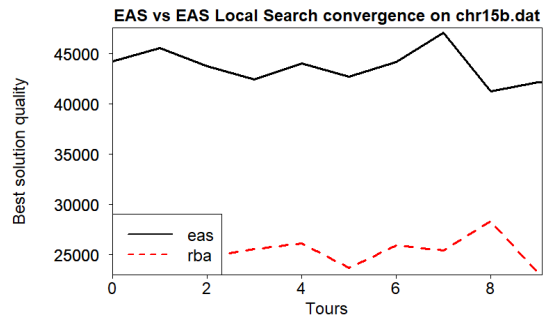


Figure 31: Results on the both algorithm on the instance chr15b

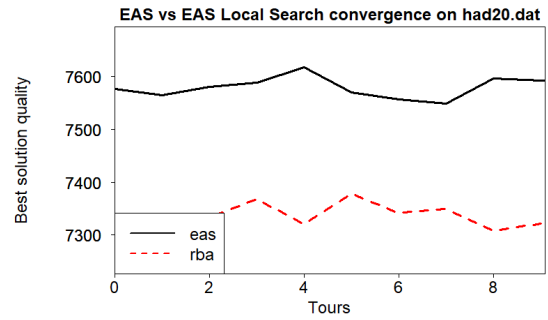


Figure 34: Results on the both algorithm on the instance had20

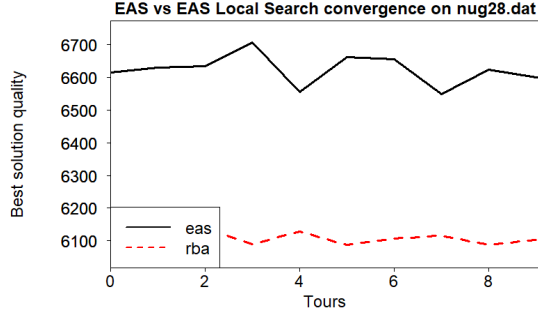


Figure 35: Results on the both algorithm on the instance nug28

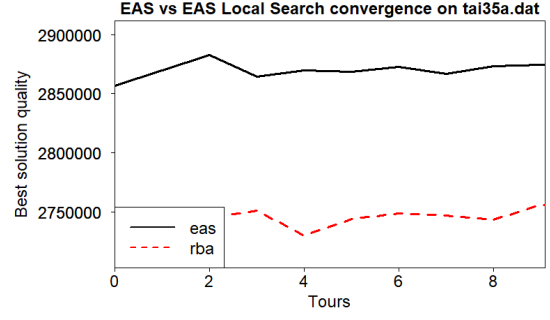


Figure 38: Results on the both algorithm on the instance tai35a

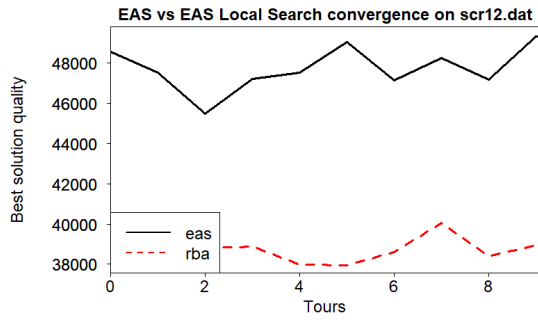


Figure 36: Results on the both algorithm on the instance scr12

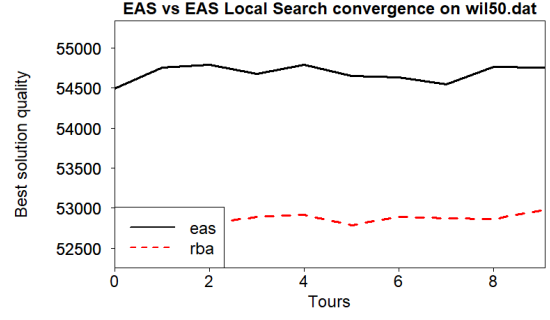


Figure 39: Results on the both algorithm on the instance wil50

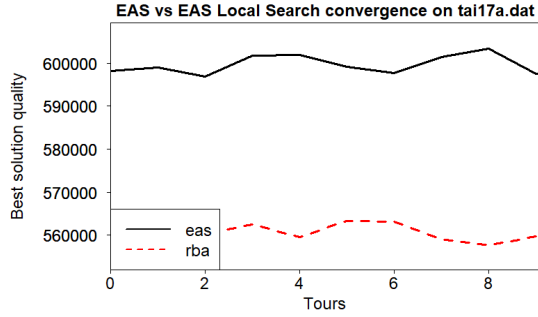


Figure 37: Results on the both algorithm on the instance tai17a

5.2 Analysis

As we can expect, the local search provides much more better results than the version without local search. The Elitist Ant system still provides good results compared to the Rank-Based Ant. When we tune the parameters of the algorithms, the results of EAS with local search don't have a really impact. The results and the quality remain almost the same. Only the behavior of the curve change a little bit but doesn't give a major difference. About the performance, the local search has a very good performance and reaches really fast the best cost.

6 Discussing and conclusion

We compared the Elitist Ant and Rank-based Ant for seeing which one give the best results

on the Quadratic Assignment Problem. We noticed that both algorithm are almost the same. Sometimes, RBA will provide a better solution than EAS. In a most cases, EAS returns the best one. About the performance, both algorithm have the same behavior. Based on the experiments, we can conclude that the best (of both) algorithm for solving the QAP is *Elitist Ant system*.

7 References

1. Frank Hutter, Holger H. Hoos, Kevin Leyton-Brown and Thomas Stutzle - *ParamILS: An Automatic Algorithm Configuration Framework*
2. https://en.wikipedia.org/wiki/Quadratic_assignment_problem
3. <http://cs.ulb.ac.be/public/teaching/infoh414>