Facultat Informàtica de Barcelona

Algorithmic Methods for Mathematical Modeling
amMM(MIRI)

AMMM Project

Authors:
Adrian Rodriguez Bazaga,
Pau Rodriguez Esmerats

January 8, 2018

Contents

Abstract

Optimization problems can appear in almost all situations on life, but there are special important cases where those problems appears, specially on the industry, since if we can achieve an optimal solution, we will improve the efficiency of the industrial process and then get lower production cost with the same (or better) efficiency. This improvement of the costs have an effect on the competitiveness of the companies and on the final quality of their products, including that this is an important money-saving factor.

The challenging part is when this kind of problems become really big, since then they have a attached really high computational complexity and cost, therefore we need some methods to face them.

To do so, we have two approaches, the always-optimal methods that obtain the optimal solution but taking into account every possible combination of the problems' solutions, as for instance Integer Linear Programming does. By the other hand we have those methods that concerns about fair execution times and are looking for a trade-off between acceptable execution times and the quality of the solution.

In this work we will propose two approaches in order to solve the scheduling of nurses in a hospital taking into account several constraints.

1 Introduction

Optimization problems can appear in almost all situations on life, but there are special important cases where those problems appears, specially on the industry, since if we can achieve an optimal solution, we will improve the efficiency of the industrial process and then get lower production cost with the same (or better) efficiency. This improvement of the costs have an effect on the competitiveness of the companies and on the final quality of their products, including that this is an important money-saving factor.

The challenging part is when this kind of problems become really big, since then they have a attached really high computational complexity and cost, therefore we need some methods to face them.

To do so, we have two approaches, the always-optimal methods that obtain the optimal solution but taking into account every possible combination of the problems' solutions, as for instance Integer Linear Programming does. By the other hand we have those methods that concerns about fair execution times and are looking for a trade-off between acceptable execution times and the quality of the solution.

In this work we propose two approaches in order to solve the scheduling of nurses in a hospital taking into account several constraints. The first one is using Integer Linear Programming and the second one by using metaheuristics. In the metaheuristics part we are focusing specifically in the Greedy Randomized Adaptative Procedure (GRASP) and the Biased-Random Key Genetic Algorithm (BRKGA).

Firstly we compare the efficiency in terms of solving time and wellness of ILP and metaheuristics over medium-sized problems.

Lastly, since large problems are intractable for ILP due to combinatorial explosion reasons, we compare the metaheuristics among them to compare such kind of problems.

1.1 Problem Statement

A public hospital needs to design the working schedule of their nurses. As a first approximation, we are asked to help in designing the schedule of a single day. We know, for each hour h, that at least demandh nurses should be working at the hospital. We have available a set of nNurses nurses and we need to determine at which hours each nurse should be working. However, there are some limitations that should be taken into account:

- Each nurse should work at least minHours hours.
- Each nurse should work at most maxHours hours.
- Each nurse should work at most maxConsec consecutive hours.
- No nurse can stay at the hospital for more than maxPresence hours (e.g. if maxPresence is 7, it is OK that a nurse works at 2am and also at 8am, but it not possible that he/she works at 2am and also at 9am).
- No nurse can rest for more than one consecutive hour (e.g. working at 8am, resting at 9am and 10am, and working again at 11am is not allowed, since there are two consecutive resting hours).

The goal of this project is to determine at which hours each nurse should be working in order to minimize the number of nurses required and satisfy all the aforementioned constraints.

1.2 Document Structure

The structure of this document is the following: The problem definition is done in chapter 2. Here it is explained the problem that is faced, which constraints is needed to take into account and what we want to optimize. At chapter 3 is explained the ILP model that has been developed, i.e. the decision variables and also the constraints with a mathematical nomenclature. After this chapter, at chapter 4 is explained how it has been used the heuristics approach in order to face with the problem, two meta-heuristics has been used: GRASP3 and BRKGA4 . After perform several executions for those approaches, a comparison in terms of time and quality of the result is done at chapter 5. Finally conclusions of the project are explained at chapter 6.

2 Integer Linear Programming

Integer Linear Programming is the first of the two methods that has been used in this project (see Introduction for more details about the problem). This method always finds the optimal solution without taking into account the amount of computational resources needed. This model is developed in CPLEX and the model implemented is described in the next sections.

2.1 Decision variables

- $w_{n,h}(\mathbb{B})$: This boolean variable specifies whether the nurse n works at the hour h (1) or not (0)
- $z_n(\mathbb{B})$: This boolean variable specifies whether the nurse n works during the shift or not.
  - ⋆ $zn = 1 \Rightarrow$ The nurse n works at least 1 hour, $\exists h, w_{n,h} = 1$
  - ⋆ $zn = 0 \Rightarrow \forall h, w_{n,h} = 0$
- $s_n(\mathbb{N})$: Positive integer variable specifying the hour in which the nurse n starts working, such that $w_{n,s_n} = 1$ and $w_{n,s_n-i} = 0$, $\forall i : 1 \leq s_n - i < s_n$
- $e_n(\mathbb{N})$: Positive integer variable specifying the hour in which the nurse n stops working, such that $w_{n,e_n} = 1$ and $w_{n,e_n+i} = 0, \forall i : e_n < e_n + i \leq 24$

2.2 Instance parameters

- $demand_h$: Array of integers, specifying the required number of nurses at hour h
- $nNurses$: Integer that specifies the number of available nurses to assign.
- $minHours$: Integer that specifies the minimum number of hours that a nurse must work if she works.
- $maxHours$: Integer that specifies the maximum number of hours that a nurse must work if she works.
- $maxConsec$: Integer that specifies the maximum number of consecutive hours that a nurse can work.
- $maxPresence$: Integer that specifies the maximum number of hours that a nurse can stay at the hospital.

2.3 Objective function

Min: $\sum_{n=1}^{nNurses} z_n$

This objective function aims to minimize the number of working nurses, which is the main goal for our problem.

2.4 Constraints

- Set the z_n values correctly:

  $\forall n : 1 \leq n \leq nNurses,$
  $24 \cdot z_n \geq \sum_{1 \leq h \leq 24} w_{n,h}$
  $z_n \leq \sum_{1 \leq h \leq 24} w_{n,h}$

- At any hour h, at least demandh nurses must be working:

  $\forall h : 1 \leq h \leq 24,$
  $\sum_{1 \leq n \leq nNurses} w_{n,h} \geq demand_h$

- Each nurse that works, must work at least minHours:

  $\forall n : 1 \leq n \leq nNurses$
  $\sum_{1 \leq h \leq 24} w_{n,h} \geq minHours \cdot z_n$

- Each nurse that works, must work at most maxHours:

$\forall n : 1 \leq n \leq nNurses$
$\sum_{1 \leq h \leq 24} w_{n,h} \leq maxHours \cdot z_n$

- Each nurse works at most maxConsec consecutive hours:

$\forall n : 1 \leq n \leq nNurses,$
$\forall h_1 : 1 \leq h_1 \leq 24 - maxConsec,$
$\sum_{h_1 \leq h \leq h_1 + maxConsec} w_{n,h} \leq maxConsec$

- Each nurse can stay in the hospital at most maxPresence hours:

$\forall n : 1 \leq n \leq nNurses, \forall h : 1 \leq h \leq 24, e_n \geq h \cdot w_{n,h}$
$\forall n : 1 \leq n \leq nNurses, s_n \geq 0$
$\forall n : 1 \leq n \leq nNurses, \forall h : 1 \leq h \leq 24, s_n \leq (h - 24) \cdot w_{n,h} + 24 \cdot z_n$
$\forall n : 1 \leq n \leq nNurses, e_n - s_n + 1 - (2 \cdot 24) \cdot (1 - z_n) \leq maxPresence \cdot z_n$

- Each nurse can rest at most one consecutive hour (exam hint version):

$\forall n : 1 \leq n \leq nNurses, \forall h : 2 \leq h \leq 23 : r_{n,h} = 1 - w_{n,h}$
$\forall n : 1 \leq n \leq nNurses, \forall h : 2 \leq h \leq 23 : wa_{n,h} = w_{n,h+1}$
$\forall n : 1 \leq n \leq nNurses, \forall h : 2 \leq h \leq 23 : wb_{n,h} = w_{n,h-1}$
$\forall n : 1 \leq n \leq nNurses, \forall h : 2 \leq h \leq 23, \forall M : M \geq 24$
$M \cdot (1 - r_{n,h}) + M \cdot wb_{n,h} - 24 \cdot wa_{n,h} + 24 \cdot r_{n,h} \geq \sum_{1 \leq h_i \leq h} w_{n,h_i}$
which is equal to :
$2 \cdot M \cdot (1 - r_{n,h}) + M \cdot wb_{n,h} - M \cdot wa_{n,h} + M \cdot r_{n,h} \geq \sum_{1 \leq h_i \leq h} w_{n,h_i}$

- Each nurse can rest at most one consecutive hour:

$\forall n : 1 \leq n \leq nNurses, \forall h : 2 \leq h \leq 22, \forall M : M \geq 24$
$M - M \cdot w_{n,h-1} + M \cdot w_{n,h} + M \cdot w_{n,h+1} \geq \sum_{h+1 \leq h_i \leq 24} w_{n,h_i}$
can be rewritten as :
$\forall n : 1 \leq n \leq nNurses, \forall h : 2 \leq h \leq 22, \forall M : M \geq 24$
$M - M \cdot wb_{n,h} + M \cdot w_{n,h} + M \cdot wa_{n,h} \geq \sum_{h+1 \leq h_i \leq 24} w_{n,h_i}$
$M - M \cdot wb_{n,h} + M \cdot (1 - r_{n,h}) + M \cdot wa_{n,h} \geq \sum_{h+1 \leq h_i \leq 24} w_{n,h_i}$
$M \cdot (2 - r_{n,h}) - M \cdot wb_{n,h} + M \cdot wa_{n,h} \geq \sum_{h+1 \leq h_i \leq 24} w_{n,h_i}$
$2 \cdot M \cdot (1 - r_{n,h}) - M \cdot wb_{n,h} + M \cdot wa_{n,h} + M \cdot r_{n,h} \geq \sum_{h+1 \leq h_i \leq 24} w_{n,h_i}$
$2 \cdot M \cdot (1 - r_{n,h}) - M \cdot wb_{n,h} + M \cdot wa_{n,h} + M \cdot r_{n,h} \geq \sum_{h \leq h_i \leq 24} w_{n,h_i}$

3 Meta-heuristics

3.1 GRASP implementation

A brief context of the GRASP:

- acronym
- building blocks / components
- intensification, diversification and strategy applied

3.1.1 Constructive phase

This paragraph shows the pseudocode of our GRASP model for the problem of interest.
After the pseudocode is shown, brief comments of each line should be added.

3.1.2 Local Search

Tests performed to determine if the model is correct. Results obtained (very brief)

3.1.3 Greedy cost function

Aab

3.2 BRKGA implementation

A brief context of the BRKGA metaheuristic algorithm:

- acronym
- building blocks / components
- intensification, diversification and strategy applied

3.2.1 Chromosome structure

This paragraph shows the pseudocode of our BRKGA model for the problem of interest.
After the pseudocode is shown, brief comments of each line should be added.

3.2.2 Decoder

Tests performed to determine if the model is correct. Results obtained (very brief)

4 Testing, Tunning and Comparisons

4.1 Benchmarking instances

This section shows how the instances have been generated. Possibly adding a very schematic pseudocode of our generation algorithm.

This section also explains the sets of instances that are prepared for the project:

- small-medium set: used to compare the solving time of the ILP model versus the Metaheuristics models
- large set: used to test the Metaheuristics models parameters over a large set of instances in order the choose the best performing setup for each model.

4.1.1 The instance generator

pseudocode
comments
how feasibility is assured
tests on how to increment solving time, including some graphs with the Best Integer and Best Bound evolution for different variations of the same instance
Conclusion on how to increase the "size" of the instance without computing the solving time of the ILP model.

The next figures show different executions of a small modification of a problem instance. In each figure, a parameter of the problem is modified to increase the time it takes to Cplex to solve the instance problem. This gives us an idea of what parameters increase the size of the problem instance.

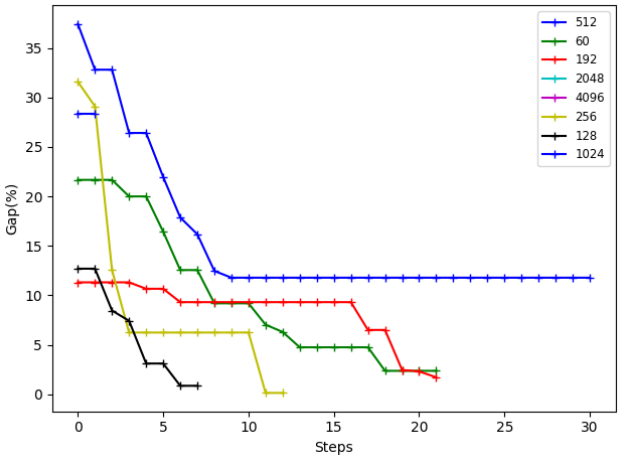


Figure 1: Evolution of Gap in similar problem instances with different number of nurses

This chart shows that increasing the number of nurses increases the number of steps of the MILP problem to solve it.

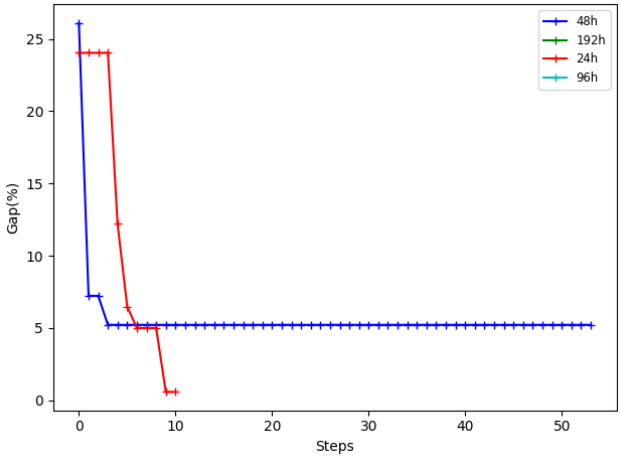


Figure 2: Evolution of Gap in similar problem instances with different number of hours

As we can observe in this chart, when the hours parameter increases, the number of steps to solve the MILP problem increases.

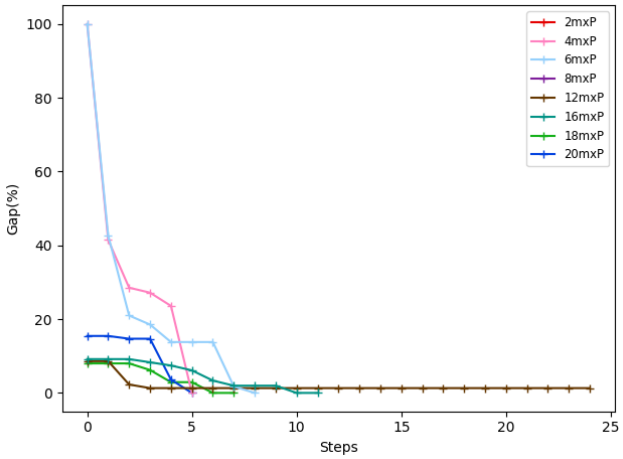


Figure 3: Evolution of Gap in similar problem instances with different values of maxPresence parameter

This chart shows that decreasing the value of the maxPresence parameter increases the number of steps needed to solve the MILP problem.

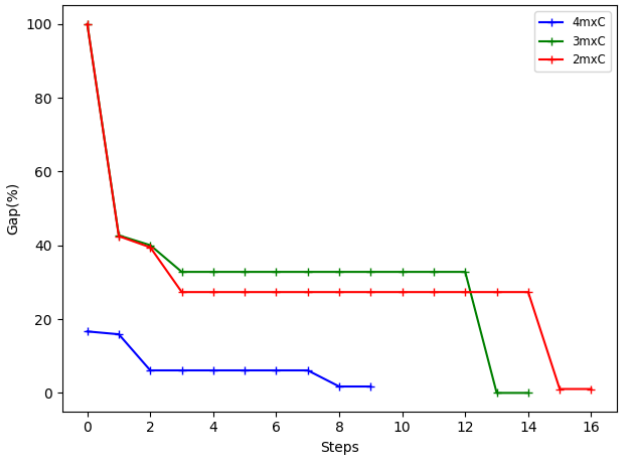


Figure 4: Evolution of Gap in similar problem instances with different values of maxConsec parameter

The chart that shows the gap and the steps of the MILP problem for similar instances with different values of maxConsec parameter, allows us to conclude that decreasing the value of the maxConsec parameter increases the size of the problem.

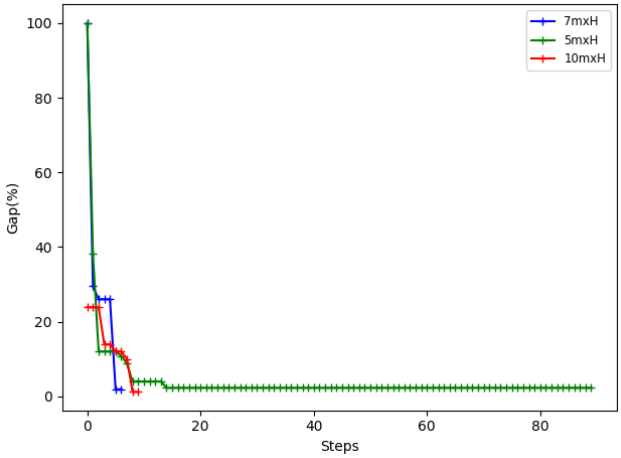


Figure 5: Evolution of Gap in similar problem instances with different values of maxHours parameter

As we can see in this chart, the higher the maximum number of hours the nurses can work, the faster the MILP reduces the gap. In that case, reducing the number of maxHours parameter increases the size of the problem.

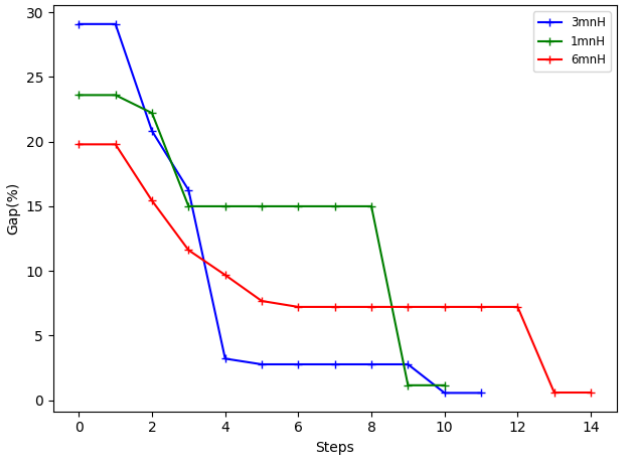


Figure 6: Evolution of Gap in similar problem instances with different values of minHours parameter

We can observe in that last chart that if the minimum number of hours the nurses must work during a shift increases then the number of steps for the MILP to reduce the gap increases. So increasing the minHours parameters increases the size of the problem.

Those small observations allow us to conclude how the parameters of the MILP problem instance can be modified to increase its solving time. Thus we can increase the size of the problem by the following modifications:

- increasing the number of nurses (nNurses)
- increasing the number of hours of the schedule (hours)
- decreasing the maximum presence hours (maxPresence)
- decreasing the maximum consecutive working hours (maxConsec)
- decreasing the maximum total working hours (maxHours)
- increasing the minimum total working hours (minHours)

The large set of problem instances will be generated using modifications of those parameters .

4.1.2 Medium Set instances

Composition of the set, number of instances and solving times.

4.1.3 Large Set instances

Composition of the set, number of instances, solving times or another measure of the size of the problem.

4.2 Comparison: ILP vs Meta-heuristics

Setup of the test:

- instances set
- execution conditions (same computer)
- models parameters
- results and times gathering procedure

Results

Chart of Solving times for all the instance set for the three models

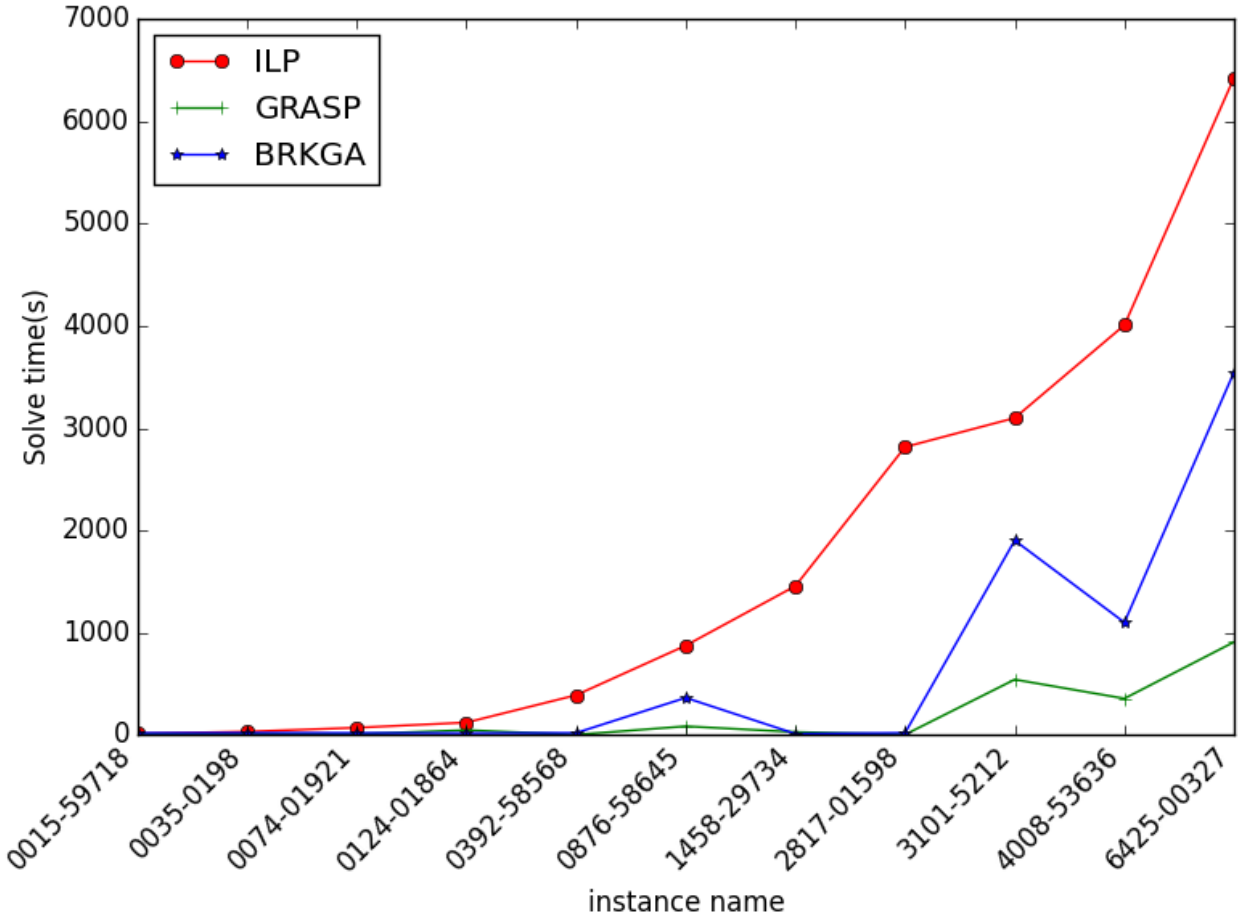
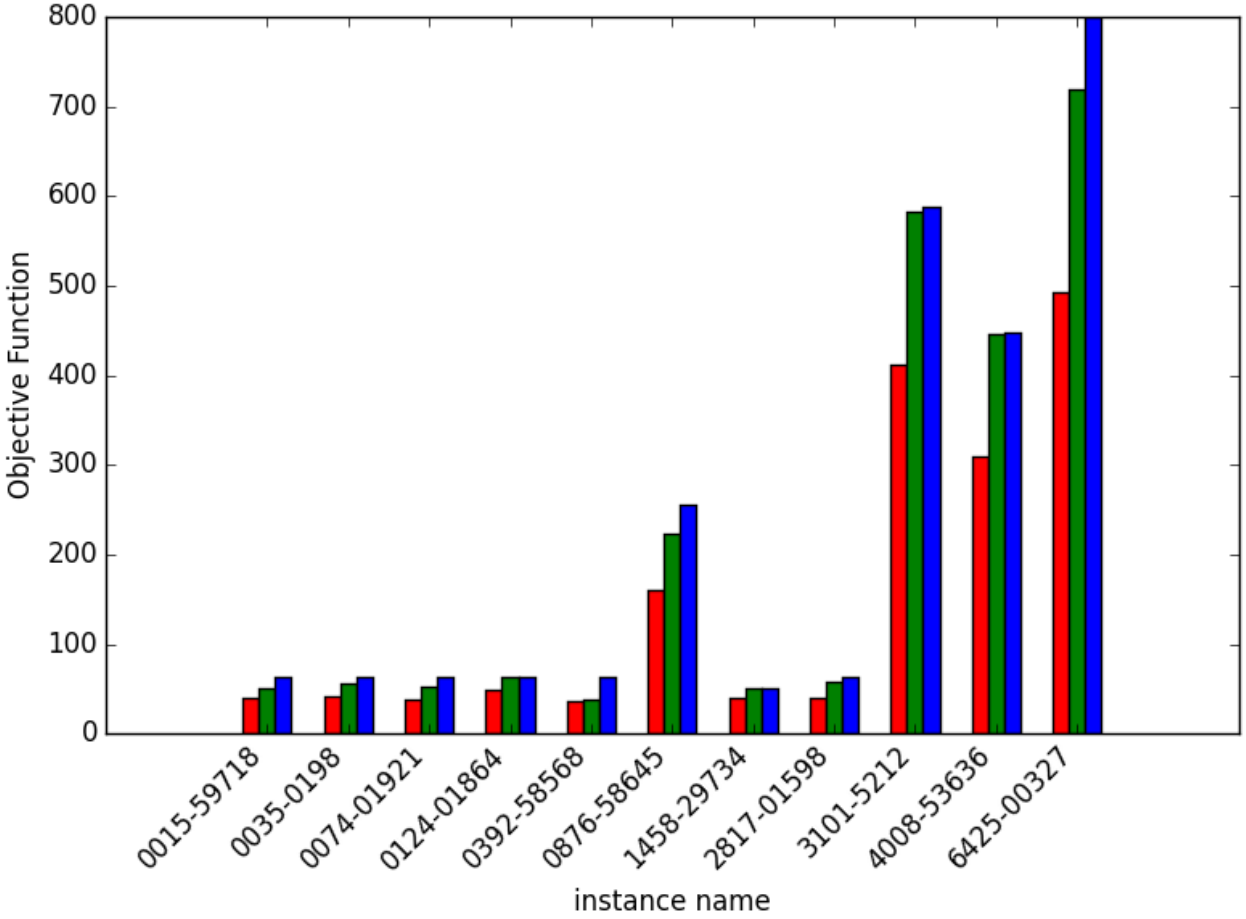


Chart of Objective function value for all the instance set for the three models


800
700
600
500
400
300
200
100
0
Objective Function
0015-59718
0035-0198
0074-01921
0124-01864
0392-58568
0876-58645
1458-29734
2817-01598
3101-5212
4008-53636
6425-00327
instance name

4.3 Comparison: GRASP vs BRKGA

Test of the section: parameter selection for each model, model performance comparision with best parameter setup.

4.3.1 Tuning GRASP parameters

Setup of the tests:

- instances set
- execution conditions (same computer)
- procedure
- results and times gathering procedure

Graph of alpha
Graph of Iterations
Graph of best improvement vs first improvement?

4.3.2 Tuning BRKGA parameters

Setup of the tests:

- instances set
- execution conditions (same computer)
- procedure
- results and times gathering procedure

Graph of iheritance probability
Graph of elite proportion
Graph of mutant proportion
Graph of population size
Graph of number of generations

4.3.3 Comparative results of meta-heuristics performance

Choosing the best performing parameter setup of the two models, perform a comparison of how objective function evolves in relation to time.

Draw conclusions about, why one is better than the other, why one stops sooner than the other. (think in terms of diversification and intensification and solution space exploration)

5 Conclusions

The conclusion may contain facts discovered throught the comparision of solving times. Thoughs about intensification and diversification of the algorithms. Parametrizations that work and why the do, etc..