Todo list

Using Optimization Techniques for Building Investments Strategies for Reducing Fraud in Electricity Distribution

Diego Luchi [*], Fábio Fabris [†], Marcos Daniel Valadão Baroni, Flávio Miguel Varejão
Núcleo de Inferência em Algoritmos (NINFA) - Departamento de Informática
Universidade Federal do Espírito Santo
Vitória, Espírito Santo, Brasil
[*] *E-mail: diego.luchi@gmail.com*
[†] *E-mail: fabiofabris@gmail.com*

***Abstract*—Electricity fraud corresponds to a major source of loss of Electricity Distribution Companies (EDCO's) in developing countries. To attenuate this, EDCO's have at their disposal several loss reduction actions. These actions must be performed to reach a given energy loss reduction goal, established by the regulatory agency. If this goal is not reached, the profit margin of the company is reduced. This work defines a formal model for the action allocation task and proposes an exact mathematic programming method for solving small instances of the problem and two heuristic methods for larger ones. We compare the techniques considering running time and solution quality.**

***Keywords*-Optimization; Meta-heuristics; Knapsack; Experimental Evaluation;**

I. Introduction

Electricity loss due to fraud is a major concern among Electricity Distribution Companies (EDCO's) in developing countries. The current estimate in our local Brazilian distribuer is that about 13% of the overall distributed electricity is lost due to fraud (non-technical losses) whereas only 7% is lost due to inherent physical phenomena involved in the transmission of electricity (technical losses) (G. A. S. And Alencastro 2012). The excessive non-technical loss represents a profit reduction for the EDCOS's and ultimately an impact on the quality of the distribution service, which in turn motivate even more non-technical losses.

In principle, EDCO's could increase the charge of consumers to pay off the loss of electricity due to non-technical sources. However, the Brazilian regulatory agency of electricity (ANEEL) dictates that only a given amount of non-technical loss may be covered by increasing the charges of non-fraudulent clients. This electricity threshold is known as *non-technical loss reducing goal*. The electricity loss above the goal cannot be paid by consumers and results in profit reduction of the EDCO's.

In practice, the existence of a non-technical loss reducing goal means that EDCO's are forbidden to charge clients more than a fixed value per Kilowatt/hour to attenuate losses due to non-technical sources. The non-technical loss reducing goal (and indirectly the maximum Kilowatt/hour charge) is established by ANEEL by studying the EDCO's profile and the historical non-technical losses behaviour of similar regions of the country. Additionally, the EDCO's are required to *reduce* the Kilowatt/hour electricity charge if they manage to mitigate electricity loss more than the established non-technical loss reducing goal. This means that there is no gain to the EDCO's if they reduce the non-technical losses more than the established goal.

Usually the non-technical loss reducing goal is given for the following three years, defining a *non-technical loss reducing curve*, so that EDCO's may plan their *non-technical loss reducing actions* for the following years. If EDCO's don't meet the imposed non-technical loss reduction curve in a given year, their profit in that year is reduced by the corresponding value of the electricity under the goal. For instance, if the regulatory agency establishes that the non-technical loss reduction goal is $40\,GW$, $50\,GW$, $60\,GW$, and the EDCO recovers $30\,GW$, $40\,GW$, $60\,GW$; $20\,GW$ won't be sold to consumers and will ultimately represent a profit loss. However, if the EDCO's recover more electricity than the established goal in a year, they are required to reduce their gains to match the profit that they would have in that year if they were exactly on the non-technical loss reduction goal.

This scenario introduces a new problem to the EDCO's: given a investment portfolio comprised of several non-technical loss reducing actions, a non-technical loss reduction curve and a yearly budget, which is the best possible allocation of actions that maximizes the return of the investment? That is, the allocation with the greatest profit or, in other words, the best *investment strategy*. For instance, if the EDCO chooses to do nothing, it would not be able to charge for some portion of the distributed electricity, since this usually costs more than performing the non-technical loss reducing actions, doing nothing would not be the optimal course of action. On the other hand, reduce the loss to a greater extent than the imposed reduction curve is not the optimal decision, since the over-reduced electricity loss does not increase the overall EDCO profit.

The current methodology used by EDCO's is to manually select the fraud reducing actions in a heuristic trial-and-error fashion. As we shall present in this work, choosing the best non-technical loss reducing actions is a complex combinatorial problem, thus the current de facto procedure hardly finds to the optimum solution of the problem. Since reducing technical losses is an important activity of EDCOS's, this motivates the use of computational techniques to find the best, or at least a good, solutions for our problem.

A. Organization

The reminder of the paper is organized as follows: Section II defines the problem of choosing non-technical loss reducing actions more formally and explains its inherent complexity. Section III is comprised of an investigation of related works in the literature. Section IV introduces our methods for solving the defined problem. In section V we present our experimental evaluation. Finally, in section VI we make our concluding remarks about the experimental results and lay ground for future work and further research.

II. Problem Definition

To apply optimization algorithms in our problem we first need to define it formally. Lets assume that we must maximize the *net present value* (the present investment return considering inflation) for a reduction plan of M years, given:

- a yearly budget for a set of L resources o_{il}, $1 \leq i \leq M$, $1 \leq l \leq L$;
- a fraud reduction goal g_i, $1 \leq i \leq M$ that represents the amount of energy loss that must be reduced;
- an *internal rate of return* r, that represents the yearly devalorization of the investment (constant for all investments and years).

We also assume that there are several actions to choose from a portfolio of actions of size N. Each action j, $1 \leq j \leq N$, contained in the portfolio has several parameters:

- the energy value v_j, that represents the value of the unit of energy for each action j in portfolio, i.e., the value of each Kilowatt of energy;
- m_j, the maximum number of times that action j may be performed, we shall refer to it as *the market* of the action;
- $u_{j,i}$ the maximum number of times that action j may be performed in the i-th year;
- $c_{j,l}$, the l-th cost associated to each execution of action j;
- $e_{j,k}$, the amount of lost energy that the action j will reduce after k years it was executed, when taken once;
- a set $D_j \subset \mathbb{N}^*$ that represents which actions must be peformed prior action j. For each action j the action $D_{j,d}$ $(1 \leq d \leq |D_j|)$ must be executed a number of times defined by
- the parameter $Q_{j,d} \in \mathbb{R}^+$.

Our objective is to find a set of values for variables $x_{j,i}$, $\forall i, j$, $x_{j,i} \in \mathbb{N}$, the number of times that the action j will be performed in the i-th year. We wish to choose a combination of values that maximizes the net present value and is under the fraud reduction goal for all years.

The constraints of the problem are the yearly budget restriction,

$$\sum_{j=1}^{N} x_{j,i} \cdot c_{j,l} \leq o_{i,l}, \forall i, l, \tag{1}$$

the market restriction

$$\sum_{i=1}^{M} x_{j,i} \leq m_j, \forall j. \tag{2}$$

the maximum number of times the action $x_{j,i}$ may be performed in i-th year;

$$x_{j,i} \leq u_{j,i}, \forall j, i. \tag{3}$$

the goal restriction

$$\sum_{j=1}^{N} \sum_{k=1}^{M} R_{i,j,k}(\bar{x}) \leq g_i, \forall i, \tag{4}$$

$R_{i,j,k}$ being the fraud reduction in the k-th year by the action j taken on i-th year, defined as

$$R_{i,j,k}(\bar{x}) = x_{j,i} \cdot e_{j,k-i+1}, \forall k \geq i \tag{5}$$

and the dependency restriction for all actions and for all years,

$$\forall j, k \quad \sum_{i=1}^{k} x_{d,i} \geq x_{j,d} \cdot Q_{j,d}, \forall d \in D_j \tag{6}$$

To introduce the objective function we must define some auxiliary concepts: The yearly total cost, C_i,

$$C_i(\bar{x}) = \sum_{j=1}^{N} \sum_{l=1}^{L} x_{j,i} \cdot c_{j,l}, \forall i, \tag{7}$$

the organically gained profit in i-th year, due to elimination of the fraud, V_i,

$$V_i(\bar{x}) = \sum_{j=1}^{N} \sum_{k=1}^{M} R_{k,j,i}(\bar{x}) \cdot v_j, \forall i, \tag{8}$$

by definition $V_i - C_i$ is the total cash flow in i-th year.

finally, we may define the objective function as:

$$max(O(\bar{x})) = max \left(\sum_{i=1}^{M} \frac{V_i(\bar{x}) - C_i(\bar{x})}{(1+r)^i} \right) \tag{9}$$

The objective function represents the net present value, that is, the sum of the yearly cash flows, $V_i - C_i$, adjusted by the internal rate of return for all years.

A. Realistic Version

To adapt the previously defined problem to the current state of our local EDCO, we simplify it in several ways:

- We consider that the yearly budgets are insufficient to surpass the goal for any given year because this is usually the case in real world scenarios.
- We consider all actions on portfolio have an positive cash flow.
- We ignore the dependencies among actions, that is, the set D_j is empty for all actions. This is necessary to simplify the analysis we perform in this work.
- We assume that both the vectors that define the costs of the j-th action (c_j) and the i-th budget (o_i) have size 1, that is, actions use resources of only one kind.
- We assume that the energy value v_j is constant for all actions.
- We assume that the value of $\sum_{i=0}^{N} u_{j,i} \leq m_j, \forall j$. Meaning that there is no yearly limit to the amount of times actions may be performed.

Terminar seo.

III. Related Work

Related work

As far as we are know, there is no literature for this specific Knapsack Problem, however, its simpler version, As the POMMK problem is a general version of the KNC problem, we may conclude that the POMMK problem is as hard as the KNC problem.

The exact formulation of the problem has shown to be quite particular and no work addressing a similar problem was found. If the devalorization of the investments is not considered ($r = 0$), the problem can be defined as a version of the Knapsack Problem. Considering just the market constraint (2), it can be classified as a simple *Bounded Knapsack Problem*. If we just consider budget constraint (1), it can be classified as a *Multi-Knapsack Problem*. If just the goal constraint (4) is considered, the problem can be classified as a *Multidimentional Knapsack Problem*. Considering just the dependency restriction (constraint 6), it can be classified as a *Partially-Ordered Knapsack Problem* (Kolliopoulos and Steiner 2002). which was demonstrated to be *NP-Complete Hard to Approximate* by Borradaile, Heeringa, and Wilfong 2009.

As POMMK has all these restrictions, it can be considered a generalization of all then. Therefore we may conclude that POMMK is, at least, as difficult as any of those problems mentioned above. The hardness of the problem implies that after a certain problem size, the exact solution won't be available in viable running times, which justifies the use of the heuristics techniques presented herein.

IV. Solving the Problem

To solve our allocation problem we have developed three methodologies: a exact algorithm using integer programming, an heuristic algorithm using the LP-relaxed version of the integer programming model with an a-posteriori heuristic allocation scheme and a

???

algorithm. In the following subsections we present each approach and comment on their particularities.

A. Exact Mathematical Programming Approach

Diego

B. Heuristic Approach - Gradient Ascent

We use a simple heuristic approach, namely Gradient Ascent (GA), using the truncated solution of the relaxed version of the linear problem (considering continuous variables) as a initial search point. Since the relaxed version of the linear problem is easily solvable and the Gradient Ascent algorithm has fast convergence characteristics, this approach is very efficient. Algorithm 2 depicts the used algorithm.

Algorithm 1 Gradient Ascent Algorithm

```
1: function GRADIENT ASCENT(Initial solution $S_{ini}$)
2:     $S_{best} \leftarrow S_{ini}$
3:     $HasImproved \leftarrow True$
4:     while $HasImproved$ do
5:         $HasImproved \leftarrow False$
6:         for Each $Neig$ in $Neighborhood(V_{best})$ do
7:             if $Evaluate(Neig) > Evaluate(S_{ini})$ then
8:                 $S_{best} \leftarrow Neig$
9:                 $HasImproved \leftarrow True$
10:            end if
11:        end for
12:    end while
13:    return $S_{best}$
14: end function
```

Algorithm 2 Tabu Search Algorithm

```
function TABU SEARCH(Initial solution $S_{ini}$)
    $S_{best} \leftarrow S_{ini}$
    $TabuList \leftarrow \emptyset$
    while $StoppingCondition$ do
        $CandList \leftarrow \emptyset$
        for Each $S_{cand}$ in $Neighborhood(S)$ do
            if $S_{cand} \notin TabuList$ then
                $CandList \leftarrow CandList + S_{cand}$
            end if
        end for
        $S \leftarrow BestCand(CandList)$
        if $Fitness(S) > Fitness(S_{best})$ then
            $S_{best} \leftarrow S$
            $CandList \leftarrow CandList + S$
            while $size(TabuList) > maxSize$ do
                $RemoveOldest(TabuList)$
            end while
        end if
    end while
    return $S_{best}$
end function
```

C. *Meta-Heuristic Approach*

Diego

V. EXPERIMENTS

A. *Experimental Setup*

All experiments were run in Intel Core i5-2310 CPU @ 2.90GHz machines with 8GB of RAM memory running Linux Mint 13. To solve the integer programming problems we have used the GLPK LP/MIP solver, version 4.43. To facilitate the job distribution we have used the HTCondor framework.

B. *Instance Generator*

To create the instances for our analysis we have implemented a random instance generator. We have tried to mimic the characteristics of real-world actions supplied by the local EDCO. Algorithm 3 defines the pseudo-code of the random instance generator.

Algorithm 3 Random instance generator

```
1:  function GENERATOR(Correlation factor $\alpha$, Number of years $M$, Number of actions $N$)
2:      $BudgetMean \leftarrow UniformRandom(700, 800)$
3:      $BudgetVariance \leftarrow UniformRandom(10, 20)$
4:      for $i = 1 \rightarrow M$ do
5:          $o_{i1} \leftarrow GaussRandom(BudgetMean, BudgetVariance)$
6:      end for
7:      for $j = 1 \rightarrow N$ do
8:          $c_{j1} \leftarrow Min(Min(o), UniformRandom(1, 100))$
9:      end for
10:     for $j = 1 \rightarrow N$ do
11:         $m_j = 2Sum(o)/Mc_{j1}$
12:     end for
13:     for $j = 1 \rightarrow N$ do
14:         for $i = 1 \rightarrow M$ do
15:             $Wji \leftarrow UniformRandom(0, 1)$
16:         end for
17:         $Wj \leftarrow ReverSort(Wj)$
18:     end for
19:     for $j = 1 \rightarrow N$ do
20:         $s \leftarrow 0$
21:         for $i = 1 \rightarrow M$ do
22:             $s \leftarrow s + W_{ji}$
23:         end for
24:         for $i = 1 \rightarrow M$ do
25:             $W_{ji} \leftarrow W_{ji}/s$
26:         end for
27:     end for
28:     for $i = 1 \rightarrow M$ do
29:         for $j = 1 \rightarrow N$ do
30:             $e_{j,k} \leftarrow Max(0, Abs((1 - \alpha)c_{j1}W_{ji} + UniformRandom(-100, 100)\alpha))$
31:         end for
32:     end for
33: end function
```

Lines 2 to 6 present the generation of the yearly budgets from a Gaussian distribution (function $GaussRandom(mean, variance)$) that in turn has its variance and mean draw from a uniform distribution of fixed parameters (function $UniformRandom(Lower\,bound, Upper\,bound)$). The parameters were inspired by real-world actions, given by the local EDCO.

Lines 7 to 12 generate the cost and market of each action, the cost comes from an uniform random distribution with fixed parameters. Note that in line 8 we use the Min function to guarantee that the action may performed at least one time. Line 11 defines the market of each action as a function of the total budget over all years ($Sum(o)$) and the cost of the action. That is, the cheapest the action the larger the market.

Lines 13 to 27 are used to build the reverse-ordered normalized weight matrix that is used to build the recuperation over the years. After line 27, the j-th line of matrix W will contain a vector of values that add to 1 and are reversely ordered, e.g., a possible W matrix for 3 years and 4 actions could be:

$$W = \begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0.8 & 0.1 & 0.1 \\ 0.6 & 0.2 & 0.2 \\ 1.0 & 0.0 & 0.0 \end{pmatrix}.$$

Finally from lines 28 to 32 the energy recuperation values are defined for each action and each year. In line 30 we may see that the use of matrix W guarantees that the energy recuperation has an decreasing tendency over the years. In this line we also present the use of the correlation parameter α, $\alpha \in [0, 1]$, the closest alpha is to one, the weaker the correlation.

The generator was implemented in the Python programming language and it is available at

URL

.

C. Time Analysis

1) The Impact of Correlation: Literature on the knapsack problem states that a strong correlation between the value of the item and its weight greatly affects the complexity of the problem. In fact, it has been demonstrated that a weak correlation among weight and value renders the complexity of the problem polynomial on the size of the problem **??**.

To test if this phenomena repeats itself in our problem when using the exact algorithm, we have tested the values $[0.0, 0.1, 1.0]$ for α, the parameter that control the randomness of the value of the action, in a number of problem sizes. When α is 0, there is a strong correlation between the cost of the action and its energy return, that is, the more expensive the action, the more effective it is. When α is 1 there is no correlation between the value of the actions and its effectiveness. When α is 0.1, there is a weak correlation between cost and value. The alpha values of 0.0, 0.1 and 1.0 are equivalent to the the following classes of problems defined by Pisinger 2005 (respectively): *subset sum instances, weakly correlated instances and uncorrelated instances.*

Figures 1 to 3 display the amount of instances that successfully executed in less than 1 hour, given value of alpha. For each cell of the figure the exact algorithm was run 10 times using random instances of the problem. The closer to black, the greater the amount of successful runs. White cells represent problem sizes that contains 0 successful runs. From the figures it is clear that the bigger the problem the likely it is that it will take more than 1 hour, also, it may be observed that the more correlated the cost is with the profit,

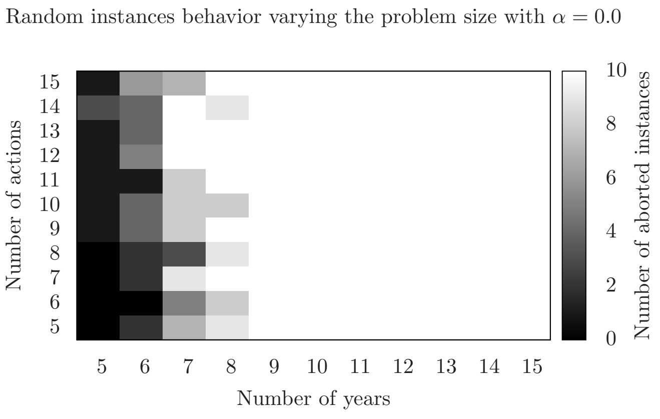

Figure 1. Strong correlation between cost and profit.

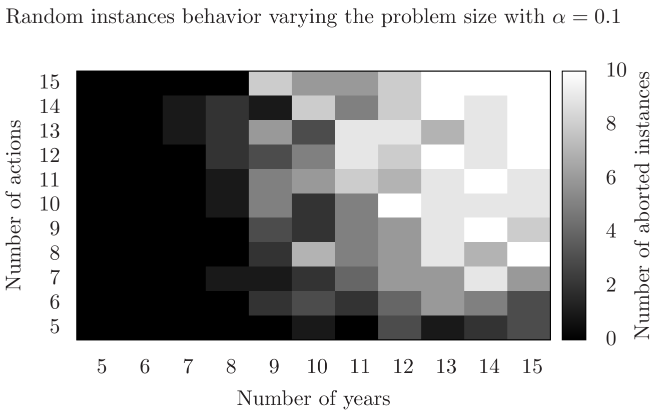

Figure 2. Weak correlation between cost and profit.

the harder the problems seems to be, as predicted by the literature.

To further investigate the impact of the value of alpha in the overall run time we have selected a cell of the figure, namely 5×5 and varied the value of alpha to analyse its impact in the running time more precisely. Figure 4 displays the impact of the value of $alpha$ in the total running time of the instances for a problem size of 5×5. The line observed in this figure suggests that the complexity of the problem decreases linearly as the correlation increases.

Figures 5 to 7 display the results for other problem sizes, it can be observed that the dimension of the problem does not alter the overall linear characteristic observed in the smaller instance. We find this result intriguing and counter-intuitive, since the existence of obviously preferable actions should

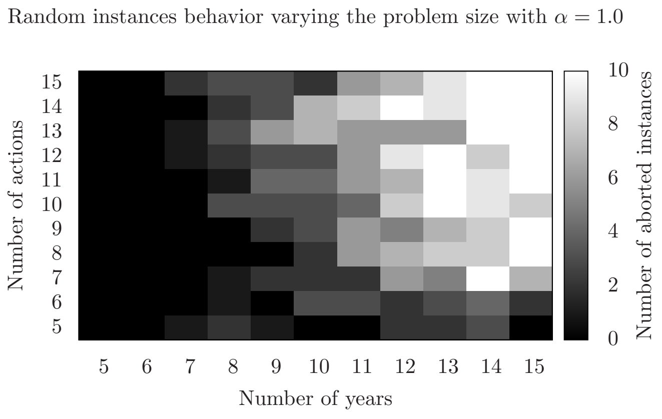

Figure 3. No correlation between cost and profit.

Figure 4. Analysis of the running time varying the value of α.

take the problem easier to solve, not the contrary.

D. Solution Analysis - Small Instances

In this subsection we compare the solution quality of the implemented heuristics in the set of small instances, there is, the instances in which the exact algorithm found the optimal solution in less then two hours. We have also limited the running time of the meta heuristics in two hours.

Figures 8 to 13 display the average ratio between the

Figure 5. Blah blah blah

Figure 6. Blah blah blah

Figure 7. Blah blah blah

optimal solution (when available) and the solution found by the meta-heuristics. White cells represent problem sizes with no optimal solution found. Small relative differences are darker than large diferences.

Discussion

Discussion

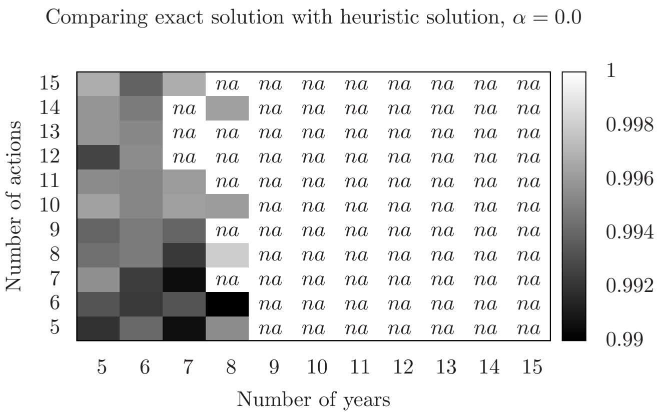

Figure 8. Comparison between the optimal solution (when available) and the solution by the meta-heuristics for $\alpha = 0.0$.

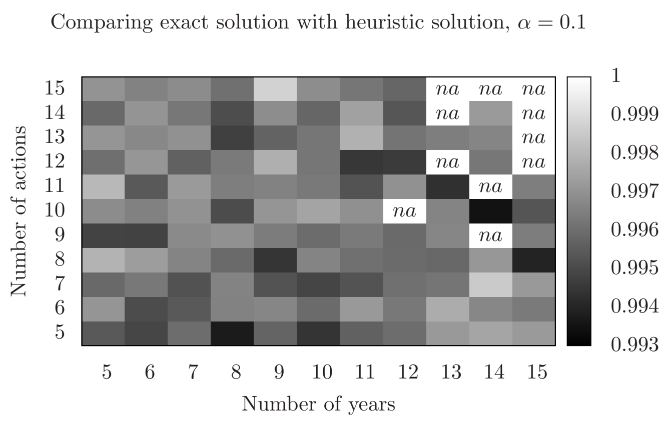

Figure 9. Comparison between the optimal solution (when available) and the solution by the meta-heuristics for $\alpha = 0.1$.

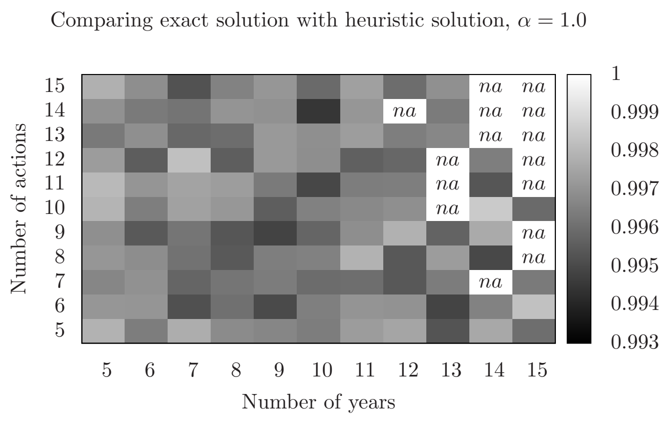

Figure 10. Comparison between the optimal solution (when available) and the solution by the meta-heuristics for $\alpha = 1.0$.

Figure 11. Blah blah blah

Figure 12. Blah blah blah

Figure 13. Blah blah blah

E. Solution Analysis - All Instances

In this subsection we compare the behavior of the meta heuristics in respect to one another, considering all instances, we cannot compare the results with the optimal solution since it is unknown for the larger instances.

Figures 14 to 16 display the relative distance between the two proposed meta heuristics considering the solution quality for the tree studied correlation values.

Discussion

VI. Conclusion and Future Work

Conclusion

Figure 14. Blah blah blah

Figure 15. Blah blah blah

Figure 16. Blah blah blah

References

[1] G. Borradaile, B. Heeringa, and G. Wilfong. "The Knapsack Problem with Neighbour Constraints". In: *ArXiv e-prints* (Oct. 2009). "arXiv": 0910 . 0777 ("cs.DS").

[2] Caputo G. A. S. and E. de Alencastro. *Nota Técnica*. 0251. SRE/ANEEL, 2012. URL: http://www.aneel.gov.br/cedoc/nreh20121326.pdf.

[3] Stavros G. Kolliopoulos and George Steiner. "Partially-Ordered Knapsack and Applications to Scheduling". English. In: *Algorithms ESA 2002*. Ed. by Rolf Mhring and Rajeev Raman. Vol. 2461. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002, pp. 612–624. ISBN: 978-3-540-44180-9. DOI: 10.1007/3-540-45749-6_54. URL: http://dx.doi.org/10.1007/3-540-45749-6_54.

[4] David Pisinger. "Where are the hard knapsack problems". In: *Computers and Operations Research* 32 (2005).