Proceedings of the 2019 Winter Simulation Conference
N. Mustafee, K.-H.G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, eds.

EVALUATION OF METAHEURISTIC ALGORITHMS FOR THE IMPROVEMENT OF SUSTAINABILITY IN THE CONSTRUCTION AREA

Lluís Aunós i Chicón

Universitat Oberta de Catalunya
Rambla del Poblenou, 156
Barcelona, CAT 08018, SPAIN

Pau Fonseca i Casas

Department of Statistics & OR
Universitat Politècnica de Catalunya
Jordi Girona 31
Barcelona, CAT 08034, SPAIN

ABSTRACT

NECADA infrastructure supports the execution of a simulation model of buildings or urban areas, taking care of environmental directives and international standards in the design process. The aim of this simulation model is to optimize the entire life cycle of the system from the point of view of sustainability (environmental, social and economic impacts), taking care of the comfort and climate change to achieve a Nearly Zero Energy Building. Due to the huge amount of factors to be considered, the number of scenarios to be simulated is huge, hence the use of optimization and specifically heuristics, is needed to get an answer in a reasonable time. This project aims to analyze the accuracy of two of the most used metaheuristics in this area. To do so we base our analysis in an extensive dataset obtained from a brute force execution, which represents a typical dataset for this kind of problem.

1 INTRODUCTION

The analysis of building sustainability encompasses the social, economic an environmental aspects, analyzing only environmental aspects, focusing on the energy demands of a building is a time-consuming task that needs resources to obtain an accurate answer. NECADA (Fonseca i Casas and Fonseca i Casas 2015) can analyze the complete life cycle of a building or a group of buildings, from its design phase to the deconstruction phase. The tool, which can be used from the cloud, includes aspects such as energy consumption, materials, design, orientation and also social repercussions. In addition, it allows simulating scenarios to know the possible effects of global warming on constructions.

The main objective of this paper is to present the analysis of the behavior of Simulated Annealing (SA) and Hill Climbing (HC) optimization algorithms in a dataset obtained from a NECADA's execution. The goal of the project where this analysis is conducted determined what are the main advantages or inconvenient of different optimization algorithms in the scope of buildings sustainability analysis. NECADA software is a Co-simulator, that can use different calculus engines to perform the calculus, in our case, the dataset is generated using TRNSYS (Thermal Energy System Specialists 2015), but other simulation engines can be used like EnergyPlus (EnergyPlus 2019), hence the accuracy of the results obtained depends on two things, (i) the accuracy of the calculus engine used, in that case is proved since we use a well-known calculus engine, and (ii) the correct definition of the experimental procedure. In this case, we base our analysis in a force-brute approach, performing an intensive calculus of all the possible scenarios, see (Fonseca i Casas et al. 2015). Hence the dataset we are going to use represents a typical case of environmental analysis (focused on energy consumption) for buildings in order to optimize the building's behavior, see (Salom et al. 2014; Fonseca i Casas et al. 2017; Ortiz et al. 2016a; Ortiz et al. 2016b) to review the context of the project where this dataset has been generated.

From this complete execution of an experimental design (brute force) obtained through NECADA, we define clearly the goal function and we develop a Hill Climbing and a Simulated Annealing for our context.

Finally, we compare the results obtained from the execution of both algorithms with the force brute solutions obtained. The language in which developments of the optimizations algorithms is made is the language R since NECADA uses R as a language for posterior data analysis; this approach is convenient for this purposes since we are focused on the accuracy of the solutions and not in the time needed to obtain these solutions. Later we can reimplement the algorithms in C to integrate them on NECADA infrastructure.

Several works in this direction have been done, analyzing the behavior of different heuristics to achieve good solutions in the frame of energy simulation for buildings. On (Evins 2013) is detailed a comparative review between different metaheuristic algorithms presenting trends and future developments in the sector. On (Suh and Park 2017) are compared a Genetic Algorithm (GA) vs a heuristic approach, concluding that the use of heuristics is interesting because of the discussion done by the stakeholders, however, metaheuristics behave better to find candidate solutions. Finally, on (Wortmann et al. 2017) are analyzed direct search, metaheuristics, and model-based methods and the performance largely depends on the parameters that are going to be used and the specificities of the problem.

2 FORMULATION OF A MULTI-OBJECTIVE PROBLEM

A Multi-objective problem is a problem of finding a vector composed by the decision variables that satisfy a set of restrictions that optimizes a set of objective functions. These functions are usually in conflict with each other, hence "Optimizing" refers to looking for a solution that owns acceptable values for all the factors to be considered. The mathematical formulation of a multi-objective problem considers the existence of several objective functions and a set of solutions. This set of solutions is based on the use of the theoretical optimality of Pareto developed by Vilfredo Pareto in 1896 (Pareto 1896).

The solution to the multi-objective optimization problem, in the sense of Pareto, will not be unique: it will consist of a set of the undominated vectors, those known as undominated set or boundary of Pareto, see Figure 1.

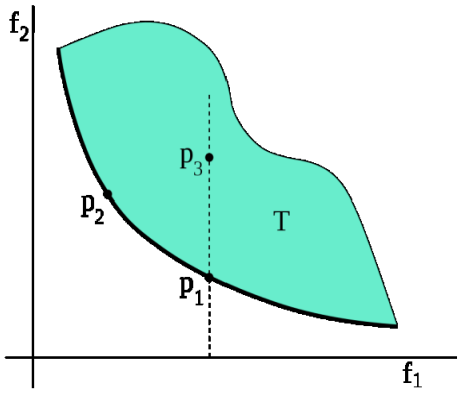


Figure 1: The Pareto frontier for a function with two objectives.

The green area T represents the solutions space for the target function. It can be observed that there is no point belonging to T that improves the solution, in the sense of Pareto, to any of the points in the border: choosing a T-point arbitrarily, for example P_3, you can trace the vertical until you get the cutting point with the Pareto frontier, in this case, P_1. This cutting point will have the same value as the f_1 And a better value of f_2. Also, you can see that for two points any of the Frontier Pareto, there will never be one that simultaneously improves the two objectives with respect to the other point of the frontier. Choosing, for example, the points P_1 and P_2, it is noted that P_1 improves f_2 (minimum is better), but at the expense of getting a worse f_1 (considering minimization case). The process of solving a multi-objective problem will be based on first, find the set of solutions that "dominate" any other solution of the solutions space and

then, select the best solution for the proposed problem, that will be that solution of the set that best suits the restrictions of the proposed problem.

3 INITIAL DATASET TO BE USED ON THE EXPERIMENT

There are 6000 simulations made with NECADA, one for each of the 6000 possible interesting combinations of building parameters (scenarios) in a common residential building in Catalonia (Fonseca i Casas and Fonseca i Casas 2015). This dataset represents the first dataset of this nature that represents the common building typologies (covering the 90% of the residential buildings in Catalonia), and the possible refurbishment measures (that one can do on this common scenarios) in order to improve buildings behavior in Catalonia. Each of these scenarios take a quite long time to run, and the analysis used to obtain the results can be considered a black-box, see (Ortiz et al. 2016a; Ortiz et al. 2016b) to review the experiment done and (Fonseca i Casas et al. 2017; Fonseca i Casas and Fonseca i Casas 2018) for the methodology used. The energy efficiency metric one can use as a reference is named LDP.

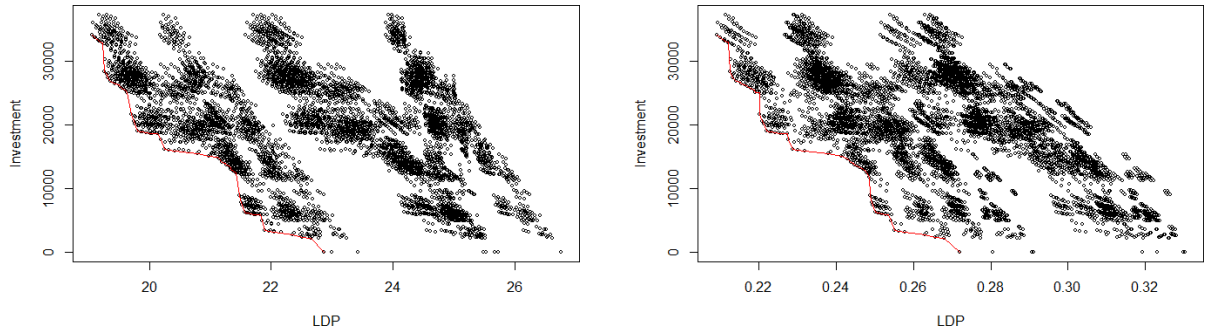
Having obtained the results for all these scenarios, we will compare now Hill Climbing and Simulated Annealing behaves to see how fast they could be to find the optimal solution to this specific problem, or in their defect how near to the optimal solution they are. This is really interesting because, considering the time needed to achieve the solution, if one wants to analyze new building typologies, as an example for isolated homes, one can rely on the metaheuristic that better behaves, without the need to use, as is done to generate this dataset, the force brute approach.

The simulations have been made through a combination of six parameters, that can own several levels, and all the possible combinations are defined through an XML file that can be understood by the simulation engine we use, SDLSP (Specification and Description Language Parallel Simulator), see (Fonseca I Casas 2010), as NECADA's core and co-simulation engine. The results that are of interest in this study are those on *Opti3Value* and *Opti2Value*, that are the two variables to improve. We can see the combination of the levels of the factors in Table 1.

Table 1: Factors to be considered in the analysis. For all the factors, the numbers represent an identifier that links with a specific level, as an example, 301 climate represents the climate file that contains the information related to Barcelona climate, for Façade, 10, 11, and so on, represent different solutions for the materials that can be used. For a deeper detail of these factors and levels (Ortiz et al. 2016a; Ortiz et al. 2016b) can be consulted.

Climate	**NVentilation**	**SProtection**	**Façade**						**Roof**			**Window**	
301, 302	0, 1	0, 1	10	11	12	13	14	15	10	11	12	10	11
			16	17	18	19	20	21	13	14	15	12	
			22	23	24	25	26	27	16	17	18		
			28	29	30	31	32	33	19				
			34										

We can define a combination (or scenario or node) of the solution space provided to us, such as a six-parameter vector (CLIMATE Nventilation, Sprotection, Façade, Roof, Window). So, for example, a possible vector of parameters or scenario, and that is one of the 6000 nodes (scenarios) that exist in the simulation results space, can be: (301, 0, 0, 10, 11, 11). If we express the results based on LDP and Investment, we obtain the map of solutions corresponding to these 6000 simulations, see Figure 2.



(a) Opti3Value (b) Opti2Value

Figure 2: Opti3Value (a) and Opti2Value (b) with the Pareto frontier drawn.

4 CODIFICATION OF THE ALGORITHMS USING R

There are many optimization functions on R, both linear and non-linear, see (Theussl et al. 2019). After analysis, we opted to do the following for the codification of both algorithms. For Hill Climbing, we do not use any features implemented already in R. The disadvantage of this is that is expected a worse performance in the execution of tests, and the advantages are that we are able to do a complete personalization of the algorithm to be used. Since we are not focused on the time performance of the algorithm, but in how close are the results from the real optimal value, this solution is good for our purposes.

For Simulated Annealing, we use the "Optim" function of the "stats" library. The advantages are the better performance in the execution and the faster deployment that offers. The disadvantages are the need to use extra mechanisms to be able to customize the different algorithm variants we want to implement, but the disadvantages are easy to get over using global variables.

4.1 "Hill Climbing" algorithm codification

We chose to implement This algorithm in R without using any of the functions that already implements it. The pseudocode is presented next:

```
currentNode = startNode;

Loop Do
    L = NEIGHBORS (currentNode);
    nextEval =-INF;
    nextNode = NULL;
    For all x in L
        If (EVAL (x) > nextEval)
            NextNode = x;
            nextEval = EVAL (x);
        If nextEval < = EVAL (currentNode)
            //Return Current Node Since Not Better Neighbors Exist Return
            currentNode;
        currentNode = nextNode;
```

The function that generates the neighborhood owns two different variants in our approach, (i) random neighborhood and (ii) consistent neighborhood.

4.1.1 Random neighborhood

Given the possible node we had mentioned previously (301, 0, 0, 10, 11, 11), a possible neighbor for this vector considering a random neighborhood is to change one-factor level with a random value within the possible values (and not the same current value). For example, if we change the fourth-factor level to 18, generates the next vector of states (301, 0, 0, **18**, 11, 11).

Because the algorithm evaluates all neighbors of a given vector, the possible neighboring vectors are all possible combinations. Evaluating all possible combinations would turn it into a thorough algorithm, which is not what we seek, and for that, we will constrain to generating as many neighbors as parameters we have. So, in every iteration we can obtain in this case 6 possible neighborhoods: (301, 0, 0, **18**, 11 11), (301, 0, 0, 10, **13**, 11), (301, 0, **1**10 11 11), (301, the **1**, 0, 10, 11 11), (**302**, 0, 0, 10, 11 11), (301, 0, 0, 10, 11 **12**). In the early implementations and testing of this algorithm, we note that the algorithm is left very early in local minimums. To avoid so, we implement a neighborhood multiplier that we can configure before the execution of tests to attempt to jump over these local minimum values and go for the global minimum values. The Default multiplier value is "6". That means, every time we evaluate neighbors, we will evaluate thirty-six neighbors (six by six multiplier).

4.1.2 Consistent neighborhood

Given the possible node we had mentioned (301, 0, 0, 10, 11, 11), the neighbors for this vector (taking into account the consistent neighborhood) would be to change the value of the parameters with value immediately previous or next to the value that it currently has. In this way, the previous vector has nine neighboring vectors:

Table 2: A consistent neighborhood for the (301, 0, 0, 10, 11, 11) scenario using a Hill Climbing approach.

Preceding	**Posterior**
(301, 0, 0, 10, 11, **10**)	(301, 0, 0, 10, 11, **12**)
(301, 0, 0, **10**, 1015)	(301, 0, 0, 10, **12**, 15)
(301, 0, 0, **34**, 11, 15)	(301, 0, 0, **11**, 11, 15)
(301, 0, **1**, 10, 11, 15)	(N/A because there are no more value options)
(301, **1**, 0, 10, 11 15)	(N/A because there are no more value options)
(**302**, 0, 0, 10, 11 15)	(N/A because there are no more value options)

That is, we generate two new neighbors, changing only one parameter, to their preceding and posterior values, except those parameters that are not possible because they only have 2 possible values, and therefore only generate a single neighbor from these parameters. If you think about changing more than one parameter at a time, the space of possible solutions to explore would be bigger and come possible variants of the algorithm. We decide to try only this variant, due to the impossibility of trying all possible variants.

4.2 "Simulated Annealing" algorithm codification

There are different alternatives in R to include functions that implement the "Simulated Annealing" (SA) algorithm like *Stats* (R Project 2018), Subselect (Orestes et al. 2018), ConsPlan (VanDerWal and Januchowski 2010) and for gradient and quasi-newton methods, GenSA (Xiang et al. 2013). We use the package «Stats", with its function "optim" to implement the Simulated Annealing algorithm more quickly. With this algorithm approach, implement three variants for the neighborhood selection.

4.2.1 Random neighborhood

Given the possible node we had mentioned previously (301, 0, 0, 10, 11, 11), a possible neighbor for this vector, is obtained changing the value of one of the parameters with a random value (avoiding the same

value). For example, if we change the fourth parameter, (301, 0, 0, **18**, 11, 11); unlike the Hill Climbing, only one neighboring node should be assessed by iteration in the algorithm.

4.2.2 Consistent neighborhood

Given a possible initial scenario (301, 0, 0, 10, 11, 11) we will obtain the same alternative scenarios presented in Table 2, but now only one alternative is going to be evaluated.

4.2.3 Consistent guided neighborhood (or using gradients)

We follow the same approach that we described previously, however, we add an observation mechanism when a parameter change causes an improvement in the evaluation result of the node. When we have obtained an improvement in this result, what we will do in the next generation is continuing to change the same factor in the same direction.
For example, from the initial scenario, we obtain the next scenario (301, 0, 0, **11**, 11, 11). From the evaluation of this scenario we obtain a better result, that the obtained with the initial scenario (301, 0, 0, **10**, 11, 11), then the next neighbor that will be generated is the following (301, 0, 0, **12**, 11, 11). This generation will cease to be guided at the time a scenario does not give an improved result.

5 OBJECTIVE FUNCTION AND EXPERIMENTS DEFINITION

The objective function wants to find the best LDP result with a budget constraint. The idea is to be able to execute the algorithms by specifying a restriction on the budget ("Investment" variables on the data), and working with two experiments, with a maximum budget of €40000 (enter all the combinations that we have), and maximum budget of €15000 (only a 26.6% of the possible value). Also, we are interested in finding the best possible savings according to a formula: Its goal is to investigate about the saving that offers to us a building, along its life cycle, considering also LDP. The objective function must ponder also other considerations like interest for the possible mortgages and others; we propose a formula to give us an idea of the goodness of the algorithms by finding a solution (1)

$$(\text{Investment} - ((\text{Ldp\_max} - \text{LDP}) * \text{Savings\_coefficient} * \text{Years})), \tag{1}$$

where:

- Investment: investment for the scenario we analyze, with constraints we discuss depending on the experiment.
- LDP _max: Maximum value of LDP found in the tests
- LDP: LDP of the scenario
- Years: 100. It is the estimated life cycle of a building.
- Savings_coefficient: 55. It is a value estimated to achieve a certain distribution in the values.

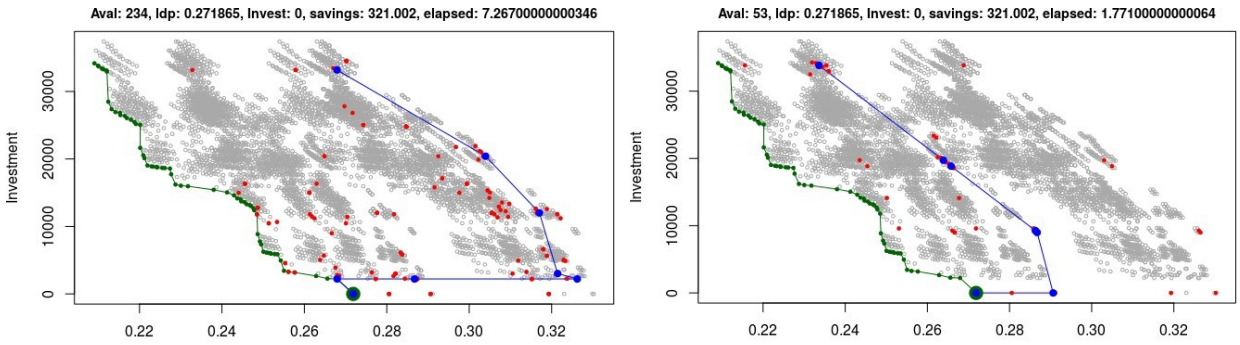
With the investment constrain and the two datasets ("Opti3" and "Opti2") we define four experiments to be used to test the algorithms. In each experiment, we will compare the algorithms to discuss the behavior depending on the constraints.

Table 3: Experiments to be executed to compare the different algorithms.

	Opti2	Opti3
€15000	HC (Random); HC (Consistent)	
€40000	SA (Random); SA (Consistent); SA (Guided)	
Best savings		

6 TESTING AND RESULTS

With each experiment, the strategy consists of four stages that are going to be executed depending on the difficult to parametrize the metaheuristic: (i) Definition of the initial configuration (parametrization) for the algorithms and execution with this configuration of an initial 10 executions, to analyze the goodness of each configuration. (ii) Reconfiguration of the algorithms based on the performance of these configurations. Execution of 10 new executions. (iii) Execution of the best configurations doing 100 executions, to confirm the consistency of the proposed configurations. (iv) Execution of the best configuration, showing the graph that depicts the search path followed by the algorithm. These graphics shows which scenarios have been evaluated of all space solutions (in red color) and which nodes have been part of the way in the search (in blue color), see Figure 3.



(a) Best value objective => Idp:0.271865, Investment: 0

(b) Best value objective => Idp:0.271865, Investment: 0

Figure 3: Examples of a Hill Climbing execution, with random neighborhood variant with 36 neighbors (a), and with a consistent neighborhood variant (b). The big dot represents one of the solutions that are considered to be implemented at the system.

The configuration for Hill Climbing is easy, and then it is not needed the four steps of the test plan explained. It is possible to start in the third step, executing a hundred of tests of each algorithm variant. With the random variant, we will be able to decide to change the multiplier for each generator and to confirm the new results. The configuration for Simulated Annealing algorithm is more complex than the Hill Climbing configuration, and therefore with this algorithm, we will execute the four phases thought of the test plan, because for each one of the algorithm variants it's needed to specify the parameters for the execution of the algorithm, the parameters are:

- TEMP: initial temperature.
- TMAX: maximum temperature.
- ITERATIONS: maximum iterations to be executed.

Configure optimally this algorithm is difficult, and there is not a rule that allows us to think in the better configuration possible. An initial idea is to test TEMP values related to the numeric differences of the results we are searching, and we want to minimize (or maximize). Then, in the first phase of the test plan execution, we will think in values that we think can be correct, but we will test other ones far away from these values to confirm that the strategy is correct. We will try reasonable values for TMAX to avoid rise so much the evaluations done. We will try not many high values for ITERATION because we don't want to do an exhaustive search. Normally, we will try values between 100 and 500. We will be able to specify different values to these parameters with global variables. Example:

- VTEMP <- c(10, 20) # Simulated Annealing temp parameter to test
- VTMAX <- c(1, 5) # Simulated Annealing tmax parameter to test

- VMAXITERSA <- c(250, 450) # num of maximum iterations for Simulated Annealing algorithm to test

The example above will execute the Simulated Annealing algorithm with eight different configurations:

Table 4: Example of multiple configurations with Simulated Annealing.

TEMP	TMAX	ITERACIONS
10	1	250
10	1	450
10	5	250
10	5	450
20	1	250
20	1	450
20	5	250
20	5	450

At the final of all tests, regardless of the number of tests that we have executed, to observe the goodness of the configuration found by each algorithm, we will output a statistic summary with the following data, see Table 5.

Table 5: Metrics to be used.

Metrics for the hypothesis of finding the best LDP with a budget constraint	
Success	Expressed with four numbers: the times that the global optimum has found (Success), those that are not (Not Success), and the percentages of success (% Success) and failure (% No Success).
LDP	Average (Mean Diff LDP), Median (MEDIAN Diff LDP), Maximum (Max Diff LDP) and Minimum (Min Diff LDP) of the difference between the LDP value of the global optimum and the LDP values of the combinations found in the tests.
Inv	Mean (Mean Diff Inv), Median (MEDIAN Diff Inv), Maximum (Max Diff Inv) and Minimum (Min Diff Inv) of the differences between the investment value of the global optimum and the investment values of the combinations found in the tests.
Aval	Average (MEAN Aval), Median (MEDIAN Guarantee), Maximum (MAX Guarantee) and Minimum (MIN Guarantee) evaluation of the objective function on the execution of each test.
Metrics for the hypothesis of finding the best savings	
Diff LDP	Average (Mean Diff LDP), Median (MEDIAN Diff LDP), Maximum (Max Diff LDP) and Minimum (Min Diff LDP) of the differences between the LDP value of the global optimum and the LDP values found in the tests.
Test differential Metrics with maximum savings.	
Diff Sav	Average (Mean Diff Sav), Median (MEDIAN Diff Sav), Maximum (Max Diff Sav) and Minimum (Min Diff Sav) of the differences between the global optimum savings and the savings found in the tests.

6.1 Analysis of the results

This discussion of the results compares both algorithms and the different strategies used to define the neighborhood based on the two separate datasets we own. We present the results in two tables, Table 6 and Table 7.

Table 6: Results for Hill Climbing in the three scenarios analyzed with the two neighborhood selection algorithms for Opti2 and Opti3 dataset.

Hill Climbing (HC)	**€15,000**	**€40,000**	**Best savings**
Opti3 HC Consistent	- Reaching local optimum with about 47 iterations on average. - finding 0% of optimum values.	- Reaching local optimum with about 57 iterations on average. - 24% of optimum values.	- Reaching local optimum with about 75 iterations on average. - 21% of optimum values.
Opti3 HC Random	- Reaching local optimum with about 135 iterations on average. - 5% of the overall optimum. - Better than Consistent: results that are near the global optimum (at the price to perform more iterations).	- 100% of optimum values with 156 iterations on average.	- 100% of optimum values with 160 iterations on average.
Opti2 HC Consistent	- Reaching local optimum with about 38 iterations on average. - 16% of optimum values.	- Reaching local optimum with about 58 iterations on average. - 24% of optimum values.	- Reaching local optimum with about 60 iterations on average. - 17% of optimum values.
Opti2 HC Random	- Reaching local optimum with about 135 iterations on average. - 59% of optimum values. - Better than Consistent: results that are near the global optimum (at the price to perform more iterations).	- 100% of optimum values with 156 iterations on average.	- 100% of optimum values with 147 iterations on average.

Results, are in general, similar for "OPTI2" and “OPTI3” datasets.

Table 7: Results for Simulated Annealing in the scenarios analyzed with the three neighborhood selection algorithms for Opti2 and Opti3 dataset.

Simulated Annealing (SA)	**€15,000**	**€40,000**	**Best Savings**
Opti3 SA Random	- 2% of optimum values with 253 iterations. - The algorithm appears unresponsive to the value of the TMAX parameter.	- 77% of optimal values with 201 iterations on average. - The algorithm seems unresponsive to the value of the TMAX parameter. - We can improve the results with more iterations, but you must	- 77% of optimal values with 201 iterations on average. - The algorithm seems unresponsive to the value of the TMAX parameter. - We can improve the results with more iterations, but you must greatly increase the

	- We can improve the results with more iterations.	greatly increase the iterations to significantly improve.	iterations to significantly improve.
Opti2 SA Random	- 46% of optimum values with 450 iterations. - The algorithm appears unresponsive to the value of the TMAX parameter. - We can improve the results with more iterations.	- 77% of optimal values with 207 iterations on average. - The algorithm seems unresponsive to the value of the TMAX parameter. - We can improve the results with more iterations, but you must greatly increase the iterations to significantly improve.	- 81% of optimal values with 251 iterations on average. - The algorithm seems unresponsive to the value of the TMAX parameter. - We can improve the results with more iterations, but you must greatly increase the iterations to significantly improve.
Opti3 SA Consistent	In all these cases: less effectiveness in finding the global optimum than with random vicinity, and higher Mean and median differences in LDP. Interestingly, with the same TEMP and TMAX configuration, we don't always remove better results with more iterations. This gives an idea of the difficulty it means to parameterize well this re-cooking algorithm Simulated Annealing.		
Opti2 SA Consistent			
Opti3 SA Guided			
Opti2 SA Guided			

Results, in general, are similar for "OPTI2" and “OPTI3” datasets. In Short, Hill Climbing seems to behave better, in this specific problem (for both datasets) when the provided vicinity is random. In addition, to achieve good results with Simulated Annealing, seems a very complicated algorithm to parametrize.

7 CONCLUSIONS

The analytic function that corresponds the results extracted by software like NECADA cannot be known are based on a simulation model that combines different simulation engines in a co-simulation approach, hence the results for each configuration are obtained through a black box. The only thing we have are the results extracted for a series of combinations of parameters, where the important data are the "LDP", which is a metric of building efficiency. Brute-force tests are costly in the execution and also in the analysis. Experimental design can be a highly ambitious project, because of the number of different parametrizations, together with the number of variants of each algorithm.

In this project, we compare Hill Climbing and Simulated Annealing algorithms using an extensive dataset extracted from simulations made by NECADA software that will represent a common dataset for the analysis of the sustainability on the construction sector.

The algorithms Hill Climbing, and the Simulated Annealing are not implemented according to a particular configuration. The possible variants to these algorithms are multiple. In this work we have worked with different variants, random, consistent and guided neighbor generation, other variants, and even the combination of both algorithms can be also considered.

Main findings are that selecting the neighborhood randomly seems to work better to achieve a result near to the optimum value. Also, the parametrization of Simulated Annealing is a key aspect (not trivial) to achieve good results respect to Hill Climbing, that is much easier to implement and configure, however, depending on the neighborhood type Hill Climbing can be stopped in a local optimum. This local optimum,

however, according to the expected results or the analysis needs, can be a good option in terms of the speed of the execution. As an example, when using the simulation engine in combination with a monitoring system where the results must be near the optimum and achieved in almost real time, hence Hill Climbing seems a good approach for this kind of problems.

This work represents an initial step and provides a methodology and a dataset useful to analyze other several metaheuristics, in order to find good candidates and parametrizations for this specific subject. This is a key aspect to obtain solutions to simulation models experimentations in almost real time, for the assessment in buildings or urban areas.

ACKNOWLEDGMENTS

The research was supported by the MED Programme of the European Union under the MARIE strategic project (agreement no.1S-MED10-002). The authors also want to thank the Catalan Housing Agency for their collaboration during the building characterization.

REFERENCES

EnergyPlus, US Department of Energy. 2019. Input Output Reference The Encyclopedic Reference to EnergyPlus Input and Output. https://energyplus.net/sites/all/modules/custom/nrel_custom/pdfs/pdfs_v9.1.0/InputOutputReference.pdf, accessed 22th September 2019.

Evins, R. 2013. "A Review of Computational Optimisation Methods Applied to Sustainable Building Design." *Renewable and Sustainable Energy Reviews* 22(1):230–245.

Fonseca i Casas, A., J. Ortiz, N. Garrido, P. Fonseca i Casas, and J. Salom. 2017. "Simulation Model to Find the Best Comfort, Energy and Cost Scenarios for Building Refurbishment." *Journal of Building Performance Simulation* 11(2):205-222.

Fonseca i Casas, P. 2010. "Using Specification and Description Language to Define and Implement Discrete Simulation Models." In *Proceedings of the 2010 Summer Simulation Multiconference, SummerSim,* 11-14 July, 2010, Ottawa, Canada, 419–426

Fonseca i Casas, P. and A. Fonseca i Casas. 2015. "NECADA. Optimization Software for Sustainable Architecture." In *Proceedings of Building Simulation Conference*, edited by J. Mathur and V. Garg, 2109-2116, Hyderabad, India.

Fonseca i Casas, P. and A. Fonseca i Casas. 2018. "Combining Formal Definition of a Simulation Model with Heuristics to Improve Building Sustainability." In *Proceedings of the 2018 Winter Simulation Conference*, edited by M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, 2375–86. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Fonseca i Casas, P., A. Fonseca i Casas, N. Garrido-Soriano, J. Aina Ortiz, and J. Casanovas. 2015. "Distributed Experiment for the Calculus of Optimal Values for Energy Consumption in Buildings." In *Proceedings of Energy, Environment, Ecosystems and Development*.edited by N.E. Mastorakis, I.R. Obuda, M.V.S. Voronezh and Y.S. Shmaliy, 41–45, Barcelona, Spain.

Salom, J., J. Ortiz, and V. Russo. 2014. "Method to Develop Cost-Effective Studies of Energy Efficiency Measures for Mediterranean Residential Existing Buildings with Multi-Criteria Optimization." In *Procediings of World Sustainable Building*. 28-30 October 2014, Barcelona, Spain, 32-40.

Orestes, J., P. Duarte, J. Cadima, and M. Minhoto. 2018. Selecting Variable Subsets. https://cran.r-project.org/web/packages/subselect/index.html, accessed 20th August 2019.

Ortiz, J., A. Fonseca i Casas, J. Salom, N. Garrido, P. Fonseca i Casas, and V. Russo. 2016a. "Comfort and Economic Criteria for Selecting Passive Measures for the Energy Refurbishment of Residential Buildings in Catalonia." *Energy and Buildings* 110(1):195–210.

Ortiz, J., A. Fonseca i Casas, J. Salom, N. G. Soriano, and P. Fonseca i Casas. 2016b. "Cost-Effective Analysis for Selecting Energy Efficiency Measures for Refurbishment of Residential Buildings in Catalonia." *Energy and Buildings* 128(1):442–457.

Pareto, V. 1896. *Cours d'Économie Politique. Tome I*, edited by F. Rouge. Genève: Librairie Droz.

R Project. 2018. "R: A Language and Environment for Statistical Computing." Core Team. https://www.r-project.org/, accessed 20th Sept 2019.

Suh, W. J. and C. S. Park. 2017. "Heuristic vs. Meta-Heuristic Optimal Energy Design for an Office Building." *Sustainability* 9(4):1-15.

Thermal Energy System Specialists. 2015. TRNSYS Transient System Simulation Tool. http://www.trnsys.com/, accessed 21st March 2015.

Theussl, S., F. Schwendinger, and H. W. Borchers. 2019. CRAN Task View: Optimization and Mathematical Programming. https://cran.r-project.org/web/views/Optimization.html, accessed 22th September 2019.

VanDerWal, J. and S. Januchowski. 2010. ConsPlan: Conservation Planning Tools. https://www.rforge.net/ConsPlan/, accessed 22th September 2019.

Wortmann, T., C. Waibel, G. Nannicini, R. Evins, T. Schroepfer, and J. Carmeliet. 2017. "Are Genetic Algorithms Really the Best Choice in Building Energy Optimization?." In *Proceedings of the 2017 Symposium on Simulation for Architecture and Urban*

Design (SimAUD 2017). Society for Modeling and Simulation International (SCS). San Diego, CA: Society for Computer Simulation International.51-58

Xiang, Y., S. Gubian, B. Suomela, and J. Hoeng. 2013. "Generalized Simulated Annealing for Global Optimization: The GenSA Package An Application to Non-Convex Optimization in Finance and Physics." *The R Journal* 5(1):13–28.

AUTHOR BIOGRAPHIES

LLUÍS AUNÓS I CHICÓN is an MS student at Universitat Oberta de Catalunya. He works at the Information Systems Department in Ajuntament de Terrassa from 1999, actually carrying out project management, service management, and technical systems management tasks. He completed studies of Techincal Engineering in Computer Systems by the Universitat Autònoma de Barcelona in 2005. His email is launos@uoc.edu.

PAU FONSECA I CASAS is a professor in the Department of Statistics and Operations Research of the Universitat Politècnica de Catalunya, where he teaches in Statistics and Simulation. He obtained his Ph.D. in Computer Science in 2007 from the Universitat Politècnica de Catalunya. He also works at the InLab FIB (http://inlab.fib.upc.edu/) as a head of the Environmental Simulation area, where he has worked on simulation projects since 1998. His research interests are discrete simulation applied to industrial, environmental and social models and the formal representation of such models. His emai is pau@fib.upc.edu.