Optimizing the cloud data center availability empowered by surrogate models

Glauco Gonçalves[1], Demis Gomes[2], Guto Santos[2], Daniel Rosendo[2], Andre Moreira[2], Judith Kelner[2], Djamel Sadok[2], Patricia Endo[3]
[1]UFRPE, Brazil - [2]UFPE, Brazil - [3]UPE, Brazil
glauco.goncalves@ufrpe.br, {demis.gomes, guto.leoni, daniel.rosendo, andre, jk, jamel}@gprt.ufpe.br, patricia.endo@upe.br

Abstract

Making data centers highly available remains a challenge that must be considered since the design phase. The problem is selecting the right strategies and components for achieving this goal given a limited investment. Furthermore, data center designers currently lack reliable specialized tools to accomplish this task. In this paper, we disclose a formal method that chooses the components and strategies that optimize the availability of a data center while considering a given budget as a constraint. For that, we make use of stochastic models to represent a cloud data center infrastructure based on the TIA-942 standard. In order to improve the computational cost incurred to solve this optimization problem, we employ surrogate models to handle the complexity of the stochastic models. In this work, we use a Gaussian process to produce a surrogate model for a cloud data center infrastructure and we use three derivative-free optimization algorithms to explore the search space and to find optimal solutions. From the results, we observe that the Differential Evolution (DE) algorithm outperforms the other tested algorithms, since it achieves higher availability with a fair usage of the budget.

1. Introduction

Cloud data center availability remains a very important provider challenge; mainly because data centers host applications with strong design requirements that often rely on always-on infrastructure, and at the same time, they manage a complex (physical and virtual) infrastructures. Failures are known to interrupt customer services, and cause financial losses for both customers and providers [1].

A typical cloud data center can be divided into three main subsystems: information technologies (IT), power, and cooling. The power subsystem is responsible for providing power to the two other subsystems. The cooling subsystem removes the heat from the premises and the IT subsystem is composed of servers, network, and storage equipment that hosts the applications and data. A failure of a component in any subsystem can affect the operation of others and, consequently, the application execution [2]. To maintain the infrastructure running is challenging due to the diversity of faults (at different levels) that may occur [3].

Fault tolerance strategies can be used to mitigate the number of failures in cloud data centers. For example, the use of redundant components is a well suited approach to mitigate faults, because when an equipment fails, another can replace it by taking over its tasks [4]. Another strategy is to invest in equipment that takes longer time to fail, i.e., equipment with a greater mean time to failure (MTTF) value. However, such hardware tends to be more expensive. Overall, the acquisition of more equipment can greatly increase the cost of running and maintaining a data center.

Thus, the planning of a high available cloud data center should take into account the acquisition cost of equipment as the constraint. This scenario can be characterized as an optimization problem [5], where the objective is to maximize the availability, considering a maximum budget that can be used to acquire equipment into the data center infrastructure. There are several algorithms used to find the optimal solution for this type of problem. In this work, we use three well-known algorithms to solve our maximization problem, namely, Particle Swarm Optimization (PSO) [6], Differential Evolution (DE) [7], and a Genetic Algorithm (GA) [8]. The goal is to compare these different strategies in terms of the availability they each achieve under given cost restrictions (equipment acquisition cost).

First, this work uses a set of Stochastic Petri Net (SPN) models (proposed by [9], [10], [11], [12]) to represent the data center infrastructure and to calculate the data center availability metric. Thus, derivative-free optimization algorithms can use such models to explore the search space and to find optimal solutions. However, the general structure of such algorithms involves solving the SPN models several times for computing availability

URI: https://hdl.handle.net/10125/63932
978-0-9981331-3-3

for each candidate solution. This leads to a high computational impact as a single SPN model can take on average about 9 hours to be solved. To overcome this computational issue, we make use of surrogate models, that "*are computationally cheaper models designed to approximate the dominant features of a complex model*" [13]. In this way, we obtain a set of equivalent mathematical functions that represent our SPN models in a reliable and faster to compute way.

This paper presents a two fold contribution: *(a)* it proposes and validates surrogate models that represent a complex SPN-based data center model; and *(b)* uses meta-heuristic algorithms to maximize data center availability (computed by surrogate models) considering the cost components' acquisition as a constraint.

The remainder of this paper is organized as follows: Sections 2 and 3 present background concepts in topics as data center modeling and optimization algorithms; Section 4 describes the need for using surrogate models to turn our problem computationally solvable, and also presents the results of experiments obtained in order to validate the surrogates models; Section 5 defines the optimization problem we want to solve and shows the results regarding the availability of data center and its cost; Section 6 depicts related work; finally, Section 7 concludes our work and delineates future works.

2. Modeling a data center

Cloud data centers are basically composed of three major subsystems: power, cooling, and IT. These subsystems are intrinsically connected, and failure in anyone can have a great impact on the other(s) [2].

Several international organizations created standards in order to define best practices and recommendations regarding data centers design and infrastructure. TIA-942 defines four tiers with different availability levels, where tier I refers to the lowest availability configuration and tier IV has the highest one. At the same time that higher tiers provide greater availability, they can be seen as being more complex due to the adopted considerable number of additional equipment associated to the different subsystems [1].

Power subsystem is responsible for providing energy to feed all data center equipment which can be divided into two parts: critical and mechanical loads. Mechanical loads refer to the energy path for cooling equipment. When this path fails, the cooling subsystem will stop working, whereas the IT equipment should continue operating until it overheats. Critical loads refer to the path that feeds the IT equipment itself. This path often comprises equipment configured to be more reliable to mitigate failures [2].

Cooling subsystem is responsible for removing the excess of heat generated by IT equipment, with the purpose of reducing damages. There are several ways to implement a cooling system in a data center [14]. However, in this paper, we consider the technique based on chilled-water only. The availability of the cooling subsystem directly impacts the availability of the IT subsystem. A failure in the cooling subsystem of a data center can result in the overheating of IT components, causing damage leading to service downtime [9].

Finally, IT subsystem is composed of servers, storage units, and a network component that connects them. The network component consists of hierarchical layers of switches, linking groups of servers where the applications are running. Technologies as network attached storage (NAS) and redundant array of inexpensive drives (RAID) connect the servers to storage units [10].

These subsystems were modeled in some previous works ([9], [10], [11], [12]) using SPN, a formalism used to model dynamic systems considering several aspects such as concurrency, synchronization, communication mechanisms, and conflicts [15]. Figure 1 shows the SPN model of a Tier I data center with its three subsystems whereas Figure 2 illustrates the SPN model of a Tier IV data center.

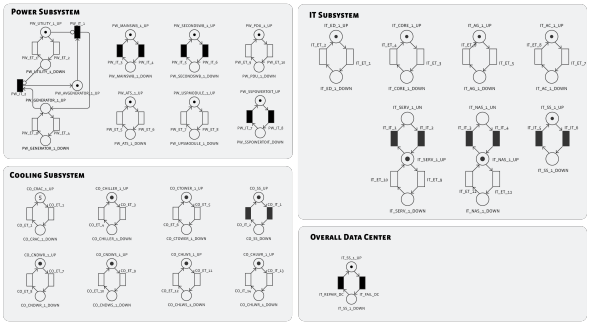

Figure 1. Tier I data center SPN model (from [16])

In a nutshell, each component of a data center is represented by a simple building block. This building block is composed of two places, which represent the status of the component (running or failed), and two transitions (that represent failure and repair events). The failure and repair transitions follow an exponential distributions, whose the averages is defined as MTTF and mean time to repair (MTTR), respectively.

Through these models, the authors have demonstrated that it is possible to calculate the availability of the data center, as well as to evaluate the cost of a given architecture, based on equipment that composes the data center. Thus, these models will be used in this work, to feed our optimization algorithms

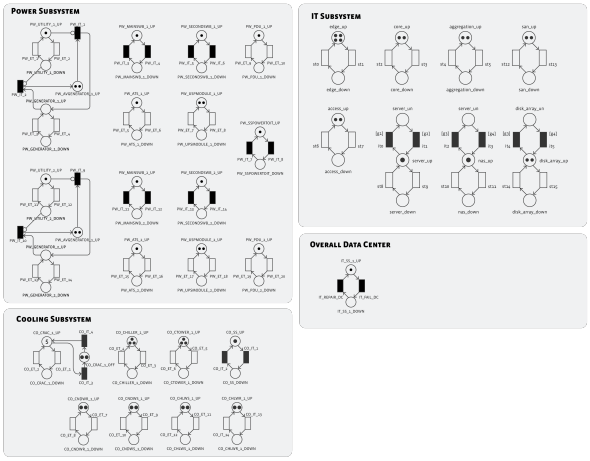

Figure 2. Tier IV data center SPN model (from [16])

in order to maximize the availability taking into account the cost as the constraint.

3. Optimization algorithms

Optimization is the process of finding best cost solution(s) to a problem, taking into account possible constraints [17]. It can be applied in several real-world scenarios: ranging from physical, time, geometric, design, etc [5]. However, finding an optimal solution is often a computationally hard task. As a result modern approaches combine meta-heuristic algorithms with some problem-specific knowledge. These algorithms are based on examining a population, where many possible solutions (population) are evaluated using a function (commonly called fitness function) in order to choose the optimal solution [17] [18].

Meta-heuristic algorithms can be grouped into evolutionary algorithms and swarm intelligence based algorithms. Among the evolutionary algorithms, one can find Genetic Algorithms (GA), Differential Evolution (DE), Bacteria Foraging Optimization (BFO), etc. Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) are examples of swarm intelligence based algorithms. In this paper, to solve the optimization problem regarding a cloud data center availability, three algorithms are applied: a GA, DE, and PSO.

3.1. Genetic Algorithms (GA)

GA is a population-based non-deterministic optimization method that emulates the selective evolution of populations for finding the optimal solution [19]. In GA, a solution is treated as a chromosome, and a set of chromosomes makes up a population. Each chromosome (a possible solution) is associated with a fitness function, to know if it represents a good result. Some chromosomes are selected by a biased random selection approach. Chromosomes with the highest values in the fitness function are selected for future iterations. The population of solutions is submitted to many biology-like operators, such as crossovers of chromosomes, the mutations of genes and the inversions of genes. The process repeats until it reaches a predetermined stopping criterion (e.g., maximum number of iterations) [19].

3.2. Differential Evolution (DE)

DE is one of the more recent evolutionary algorithms and presents a simple and efficient technique. Its simplicity is due to the fact that it requires few parameters, has a good convergence, and has functions with simple and inexpensive arithmetic operators [20]. DE is very similar to GA and is composed of four basic steps: initialization of population, mutation, crossover, and selection. The three last steps are repeated in the subsequent DE iterations [21].

An essential difference between DE and GA concerns the selection operations that they employ. In GA, the chance of a solution being selected as a parent depends on the fitness value of this solution. On the other hand, in DE, all solutions have the same chance of being selected as parents, i.e., the chance does not depend on fitness values. After the creation of a new solution using self-adjusting mutation and crossover operation, it competes with its parents, and the best one goes to the next generation [22].

3.3. Particle Swarm Optimization (PSO)

PSO is an optimization algorithm that takes advantage of the swarm intelligence mechanisms and has been widely adopted for dealing with population-based optimization problems. This algorithm is biologically inspired from flock of birds behavior, where a set of solutions to a problem (swarm) is called population [23]. A population is a set of parameters and represents a point in solution space of problem.

A PSO algorithm starts with random particles in a solution space, and each particle begins with its associated position and velocity. Then, the particles move through a solution space with the purpose of finding the optimal solution. The position and the velocity are updated based on a particles own experience and that of its neighbors. Also, the positions are distinguished as global best and personal best, based on the fitness function. Each movement performed by

particles is entirely influenced by its current position, its parameters, and group knowledge of the swarm [23].

4. Using surrogate models to reduce complexity

The data center model proposed previously, as described in Section 2, has three subsystems with many places, transitions, and tokens, which turns it a complex model to compute.

We have noted that when we increase the tier level, i.e., improving the subsystems component redundancy, the Markov chain used to solve analytically the SPN sub-model [24] also increases. For instance, when we performed stationary analysis of the models, the Markov chain related to the power subsystem path to the IT subsystem presented 32 states when considering the tier I, 84 states for tier II, 1344 states for the tier III, and the Markov chain for tier IV had as many as 7056 states. In addition to state explosion, when we performed stationary simulation in an Intel Core i7 3.47GHz and 24 GB RAM, the time to solve the model also increased: whereas tier I spent around 1 minute and 36s to solve, tier IV spent 9 hours and a half.

This situation worsened when we integrated the submodels. For instance, in [11], when the authors integrated the tier I SPN models of power and IT subsystems, the correspondent Markov chain presented 2048 states; while the integration of tier IV models suffered from the state-space explosion issue, which prevented us from exploring stationary analysis. Solving the SPN models through simulations, tier I (Figure 1) took about one hour to finish, whereas tier IV (Figure 2) took almost four days to finish.

Therefore, in order to apply optimization algorithms for such data center availability models, it is first necessary to reduce their complexity because these algorithms commonly require thousands of function evaluations to reach a solution. Using original SPN models with optimization algorithms is therefore impracticable. In this context, surrogate models can be applied to represent expensive computational models [25], once they require fewer observations than the original model and produce estimations with high accuracy. For example, it is possible to generate a surrogate model with only 30 availability values (solutions) from the original complex SPN model.

4.1. Surrogate models

Surrogate models (metamodels or approximation models) are powerful statistical methods to emulate the output of complex models [26]. There are several techniques to build surrogate models. Some of the best known are: polynomial regression, neural networks, Radial Basis Function (RBF), and Gaussian process.

Since we have complex and computationally expensive SPN models, the best technique for this work consists of building a surrogate model with a small training set. Neural Networks and RBF need a large training set. Gaussian processes do not need a huge data set and produce better results than polynomial regression [27].

This work uses a Gaussian process called design and analysis of computer experiments (DACE), also referred as kriging [28]. DACE proposes performing physical experiments, and is widely used in the computer experiment domain, it finds a spatial function that is able to replicate the output of the original model and to predict unknown points based on known values.

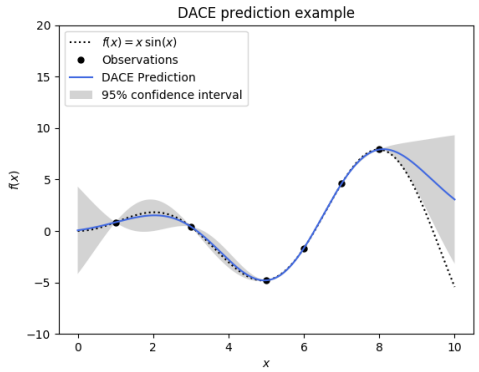

Figure 3. Example of a DACE model (based on [29])

Figure 3 shows an example of DACE. DACE uses some observations from the original model and estimates the unknown points through variance [30] between consecutive points and, hence, the error is minimal when the points are very close to observations. DACE needs few observations to generate a surrogate model with higher accuracy than other methods, such as RBF or polynomial regression.

However, as shown in Figure 3, the sampling is essential to ensure DACE accuracy. For example, when $8 < x < 10$, there are no observations, which increases the variance and, consequently, decreases the accuracy. Also, to obtain a good DACE model, the observations must be at a reasonable distance among them, which increases the chance of covering the whole original model area. A possible strategy is using Monte Carlo Simulations (MCS), but it needs a high number of samples to reach a reasonable accuracy. In this work, we adopt the Latin Hypercube Sampling (LHS), a widely adopted sampling strategy that divides the range of each variable into disjoint intervals of equal probability,

Table 1. RMSE of three different configurations of DACE model for tiers I, II, III and IV

Tier	Orig. Av. (%)	C1	C2	C3
Tier I	99.1756	3.12E-02 (+- 1.16E-04)	7.26E-03 (+- 2.70E-05)	7.40E-03 (+- 2.76E-05)
Tier II	99.8672	8.90E-03 (+- 3.32E-05)	2.16E-03 (+- 8.04E-06)	2.22E-03 (+- 8.18E-06)
Tier III	99.9477	8.36E-03 (+- 3.12E-05)	9.06E-04 (+- 3.36E-06)	9.72E-04 (+- 3.57E-06)
Tier IV	99.9489	8.36E-03 (+- 3.12E-05)	9.16E-04 (+- 3.39E-06)	9.80E-04 (+- 3.60E-06)

choosing one value of each interval [31]. Therefore, LHS allows a considerable area coverage and does not select closer observations among them.

4.2. Validation and discussion

DACE must ensure a low error in comparison with the original model. In other words, the surrogate model must obtain accurate results with the short training set to accelerate optimization process.

In order to choose the configuration with lower error, tests were carried out with different parameters to generate DACE models from the training set. The changes affect the covariance, used by DACE to estimate unknown points, and the optimization method.

We defined three configurations to evaluate the surrogate model: C1, with exponential covariance, optimization via genetic algorithms with a maximum of 1000 generations; C2, with similar configuration, only changing the covariance to Gaussian; and C3, with Gaussian covariance and Broyden-Fletcher-Goldfarb-Shanno (BFGS) optimization method [32], with 1000 start points. Comparing C1 to C2 one can observe the impact of the covariance function on the error, whereas comparing C2 to C3 one can see the impact of changing the optimization algorithm.

After generating DACE models from the training set, the models may be assessed predicting the availability for samples from the testing set and calculating the RMSE. Table 1 shows the RMSE of three configurations with 95% confidence interval and the availability of the original model. We can see that the C2 configuration had the best results, affecting the availability in lower decimal places. However, there is no statistical difference between it and the C3 configuration.

Despite the original SPN model complexity, the short training set generates DACE models with reasonable accuracy. The worst results belong to models with exponential covariance. Due to high exponential variance (coefficient of variation is 1) the DACE model estimates less confident results in unknown points, which reduces its accuracy. The result of C1 in comparison with C2 is four times less accurate in tiers I and II, and nine times less accurate in tiers III and IV.

Table 2 shows the RMSE of C2 configuration concerning downtime generated by the original model. The impact of C2 RMSE is very low on downtime, once this RMSE increases or decreases the downtime in 42 min, 13 min, 9 min, and 6 min in tiers I, II, III and IV, respectively. In tier I, for example, the downtime reaches three days, i.e., an error of 42 min is almost negligible (about 0.9% of total). Tiers II, III, and IV have similar results, the impact on downtime is, respectively, 1.8%, 3%, and 3.1%.

Table 2. Relation between downtime and RMSE of DACE model

Tier	Original Availability	Downtime (h)	RMSE C2 Downtime (h)
Tier I	99.1756947123	72.2091432	0.69
Tier II	99.8672202954	11.63150212	0.21
Tier III	99.9477237852	4.579396416	0.14
Tier IV	99.9489709783	4.470142301	0.14

5. Maximizing cloud availability

Considering the development of accurate surrogate models, it is now possible to apply the optimization algorithms in order to find solutions that maximize the data center availability considering the financial cost of the components as a constraint. In this next phase of the research, we describe the problem definition and the results we obtain from optimization algorithms that were applied on surrogates models.

5.1. Problem definition

We assume that the cost function of each data center component ($C(MTTF_c)$) follows a positive exponential relationship that varies with the correspondent mean time to failure ($MTTF_c$) of the component c. This is a reasonable assumption and the associated relationship can be modeled as

$$C(MTTF_c) = MTTF_c.e^{(f_c \times MTTF_c)} \quad (1)$$

The parameter f_c is defined using the cost of the component C_c, that may be estimated after market research, and the respective MTTF value $MTTF_c$, that is acquired from data sheets and papers. The value of f_c is given by

$$f_c = \frac{\ln(C_c) - \ln(MTTF_c)}{MTTF_c} \quad (2)$$

which is a direct result from Eq. 1.

When f_c is negative, the exponential model drops for high $MTTF_c$ values. In this case, one can use other increasing functions. In this work, in such cases, we adopt a simple linear model

$$C(MTTF_c) = MTTF_c \times f_c \quad (3)$$

where f_c is given by

$$f_c = C_c/MTTF_c \quad (4)$$

These functions will be used to calculate the cost of each component when the optimization algorithms vary the respective $MTTF_c$ values.

Thus, the optimization problem to be solved can be stated as

$$\max Availability(MTTF_1, MTTF_2, ..., MTTF_n)$$
$$\text{subject to} \quad (5)$$

$$\sum_{c=1}^{n} C(MTTF_c) \leq B \quad (6)$$

$$lb_c \leq MTTF_c \leq ub_c,\ c = 1, \ldots, m \quad (7)$$

where B is the budget available for optimization of the the financial constraint, and lb_c and ub_c are, respectively, the lower and upper bounds on the correspondent component c.

The function *Availability (MTTF_1, MTTF_2, ..., MTTF_n)* is given by the surrogate model of the data center, which is computed following the strategies discussed in Section 4. In this way, the algorithms can solve the optimization problem evaluating an accurate and fast to compute approximation of the SPN availability model.

5.2. Optimization algorithms

In this paper, we apply three optimization algorithms to solve our maximization problem: GA, DE, and PSO. These algorithms are well suited to our problem, as they converge to a stable solution, ideally the optimal solution after a certain number of iterations [33].

In order to compare the optimization algorithms, a set of the specific parameters of each algorithm is required. The PSO and DE algorithms' parameters are based on [34] and [35], while GA parameters have been selected experimentally.

With GA, the population size is 170, the crossover probability is defined as 90%, the mutation probability is 20%, the number of iterations are selected as 5,000. For DE, the crossover probability is 95.55%, the step size is 0.6497, the population size is 37, and number of generations is 40,000. In PSO, the inertia weight is -0.4438, the standard deviation of the initial velocities is 2, the population size is 170, weight towards the individuals best solution is -0.2699, the weight towards the populationss best solution is 3.3950, and number of generations is 40,000.

Next, the LHS is selected for generating the search space used in the algorithms. Search space consists of the MTTF values of the components of the data center, and the relation between MTTF and component cost described previously. All MTTR values are fixed, as shown in Table 3, assuming that the MTTR does not affect the acquisition cost of component because MTTR depends on maintenance strategies used in a company (time for fault finding and the time spent repairing) [36]. For simplicity, we focus only on tier I models for the data center.

Table 3. MTTR values of data center components (from [37], [38], [39], [40], [41],[38], and [42])

Data center component	MTTR (in hours)	Data center component	MTTR (in hours)	Data center component	MTTR (in hours)
Utility	0.03	CRAC	8	NAS	12
Generator	3.9	Chiller	48	Aggregation switch	0.63
ATS	5.74	Cooling Tower	48	Access switch	0.35
UPS	8	Pipes	2.71	Core switch	0.78
PDU	156.01	Server	0.59	Edge switch	1

The evaluation of the algorithms considered some values for the available budget set to acquire new components to the data center to maximize the availability. From the data obtained from LHS,i.e. the MTTF values of the data center components, it is estimated that the minimum and the maximum budgets are $189,972.60 and $441,227.90, respectively. Thus, the budget value varies from $200,000.00 to $400,000.00 in the experiments, ranging in steps of $50,000.00.

5.3. Results

Figure 4 shows the total cost (based on a maximum budget) that each algorithm estimated for maximizing the data center availability. In most cases, the budget constraint was respected by the optimization algorithm. Only when the budget was $200,000.00, PSO found a higher value, $226,518.60. When the budget was $350,000.00, both DE and GA achieved lower costs, $339,223.80 and $336,898.8.80, respectively; and for

this same budget, PSO reached the highest cost: $345,168.30.

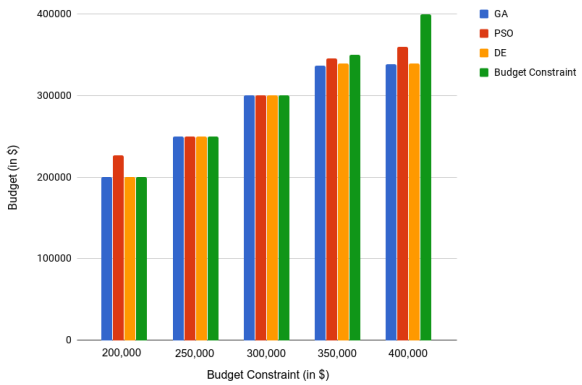

Figure 4. Budget estimated by optimization algorithms

For the higher budget value ($400,000.00), the optimization algorithms estimated a much lower value for the cost. DE, GA, and PSO estimated the costs values to $339,223.00, $338,461.80, and $359,916.30, respectively.

Figure 5 shows the availability found by the three optimization algorithms. In general, DE obtained the best availability results, followed by GA, and PSO. Only with a lower budget ($200,000.00), PSO obtained a result better than the one for GA - 99.0569% and 99.0312%, respectively. For this budget, DE found 99.10102% to be the optimal solution. With the increase of the budget, it is possible to notice that GA obtains better availability results than the PSO and is very close to the DE, which achieves the best results. With the highest budget ($400,000.00), GA, DE, and PSO found very close budget values, 99.3467%, 99.3482%, and 99.3460, respectively.

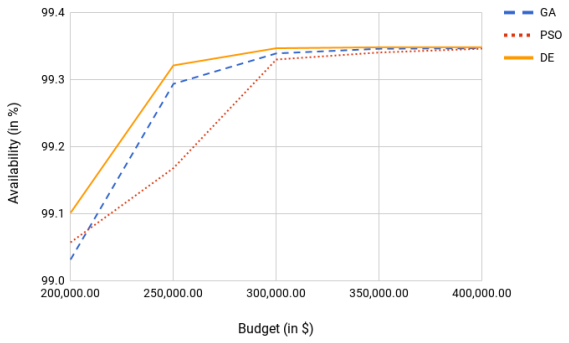

Figure 5. Availability found by optimization algorithms

5.4. Discussions

With regard to the budget estimated by the above three optimization algorithms, in general, and as expected, when we increase the value of budget constraint, there is an increase in the budget estimated by the algorithms. We had an exception when the budget value was $200,000.00; the PSO extrapolated and estimated a larger investment of $226,518.60, in other words it overshot the constraint by more than $26,518.60. Hence, in this specific case, the PSO did not find an optimal (acceptable) solution. On the other hand, when the budget was set to $200,000.00 and $300,000.00, the algorithms estimated budget values closer to the maximum. However, when we considered a top budget of $350,000.00, GA and DE estimated costs a little below this limit with, $336,898.80 and $339,223.00, respectively; while PSO found the biggest value $345,168.30. When the budget constraint is set equal to $400,000.00, all optimization algorithms estimated budgets that were lower than the budget constraint. These values are due to the fact the algorithms found an optimal local solution to the availability metric with respective budget values.

Regarding availability results, in DE, an increase of budget from $200,000.00 to $300,000.00, resulted in an increase in availability from 99.1010% to 99.3468%, showing a difference of 0.2458%. While the increase in budget from $300,000.00 to $400,000.00, increased availability from 99.3468% to 99.3482%, giving a small gap of 0.0014%. Hence, availability tends to stabilize, and budget increases lead to a very low increase in the availability of budgets above $300,000.00. Even if we increase the budget to buy additional data center equipment, increasing availability will be minimal, and perhaps not worth investing more money.

Overall, the PSO achieved the worst availability results compared with GA and DE. Only with a budget of $200,000.00, PSO found greater availability than GA and DE, 99.0569% and - 99.0312%, respectively. However, PSO extrapolated the constraint and estimated a $226,518.60 budget. As a result, in this case, PSO did not find the optimal solution. Nevertheless, DE found the highest availability for this budget, 99.1010%. With budget values from $250,000.00 to $350,000.00, all three algorithms found high availability values while fulfilling the cost constraint.

With the highest budget value, $400,000.00, DE reached the same availability results as those with a budget of $350,000.00, 99.3482%. Probably DE found an optimal local, because it obtained the same previous results, even though it is possible to increase the MTTF values and achieve greater availability since a larger

Table 4. MTTF (in hours) values found by each algorithm with the value of the budget equal to $400,000.00

	Utility	Generator	ATS	UPS	PDU	CRAC	Chiller	Cooling tower
GA	285.6947	9,041.9984	122,164.8677	24,625.8113	349,194.0399	45,519.5737	22,227.9684	30,877.1837
DE	285.7010	9,031.8223	**122,213.7580**	24,634.0189	**351,069.1174**	**45,794.5268**	**22,249.5083**	**30,992.9884**
PSO	**287.2192**	**9,805.7563**	111,445.3339	**30,900.3677**	346,989.2875	45,447.7926	**22,249.5083**	30,992.8577

	Pipe	Server	NAS	Aggregation switch	Access Switch	Core Switch	Edge Switch
GA	548,869.5849	1,451.6996	1,433,657.8275	**8,650.5430**	13,989.3466	**18,473.8811**	984.9421
DE	550,511.9486	**1,458.2706**	1,434,069.4316	8,614.4269	**14,031.5045**	18,386.3,288	**989.2192**
PSO	**552,403.0638**	1,458.2606	**1,436,703.3775**	8,242.4303	13,925.2877	15,872.3488	**989.2192**

budget value is available. GA had a little improvement in availability when compared with the $350,000.00 budget, from 99.3460% to 99.3470%, with the budget estimates of $336,898.80 and $338,461.80 repsectively.

Table 4 shows the MTTF values of the data center components achieved by each optimization algorithm for the highest availability value found (budget equals to $400,000.00). Cells with light gray color show the lowest MTTF value compared with other algorithms, while cells with dark grey color show the highest value.

Considering the MTTF obtained by each data center component, GA found the lowest value eight times, and the highest value twice. Thus, GA tried to considerably increase MTTF of few components, while it increased MTTF of other components slightly, with the intention of not extrapolating the budget.

Considering the MTTF of data center components, DE found the lowest MTTF value once, and the highest value seven times. The strategy used by DE is to improve several components to increase the availability, but probably the chosen components did not have a high cost to extrapolate the budget constraint.

Finally, PSO found the lowest MTTF value six times, and the highest value five times. PSO estimated the highest budget with the lowest availability. Thus, the components that PSO increased their MTTF values did not have much impact on availability, even though they had raised the cost. PSO greatly increased the MTTF values of generator and UPS when compared to GA and DE. These components are the most expensive of the power subsystem, which justifies the high budget estimated by the PSO.

Considering the number of components per subsystem, with the highest MTTF values found, the DE focused on cooling and IT subsystems (i.e., it searched for optimum results by changing MTTF values of those subsystems' components), while PSO focused on power subsystem, and GA invested equally. DE increased the MTTF of the cooling and IT components and obtained the best results of availability. However, PSO tried heavily to find optimum results by changing the power components' MTTF values (mainly generator and UPS), while neglected the other subsystems, and thus obtained the worst availability result. The GA sought to optimize equally, looking into all subsystems' parameters, and thus obtained near availability results compared to the DE.

6. Related work

In [43], authors deal with the optimization problem of a Web service workload assignment in a cloud infrastructure scenario. They proposed a framework that aims to minimize cost and maximize resource utilization by determining the optimal mapping of virtual machines to physical machines in a cloud data center. The Integer Linear Programming (ILP) approach was used to optimize the server workload assignment. Their results presented an annual reduction in the per hour on-demand cost of the instances by 52%. Moreover, the resource utilization results showed improvements in CPU and memory utilization by 20.66% and 13.42%, respectively.

A set of models to accurately predict the energy consumption in Cloud data centers was proposed in [44]. Their models were built based on Principal Component Analysis (PCA) and regression methods such as linear regression, power regression, exponential regression, and polynomial regression. The regression models were compared to real energy consumption values measured using Power Bay-SSM tool and compared to existing approaches such as Ramon Model, Linear Model, and Cubic Model. Results show that, after a comparison between the seven approaches, the power regression models offer the highest accuracy.

In [45], authors propose a methodology for online optimization of cloud data center configuration. The configuration includes the number of active VMs, CPU utilization, resource requests, and CPU demand per request. Surrogate models based on statistical regression techniques were applied to predict the quality of cloud configurations. The CloudSim simulation tool was used in evaluation, and their results showed that the surrogate models produced high-quality configurations

with a prediction error within 6%.

The problem of service allocation in cloud computing is handled in [46]. The proposed solution aims to maximize revenue for users and providers and find the optimal solution. Multi-Objective Particle Swarm Optimization based on Crowding Distance (MOPSO-CD) was used to solve the service allocation optimization problem. Authors compared the performance of their approach against Non-dominated Sorting Genetic Algorithm II (NSGA-II) and Multi-Objective Particle Swarm Optimization (MOPSO) using MATLAB. Results show that MOPSO-CD improves the execution time of the resources allocation algorithm while generating high revenue for both users and providers.

Differently from these previous works, we propose and validate surrogate models based on the DACE method to assess the availability of different cloud data center architecture configurations (using models proposed by [9], [10], [11], [12]). Furthermore, we apply and compare the accuracy of three optimization algorithms (PSO, DE, and GA) to maximize the data center availability given different budget constraints.

7. Conclusions and future works

Planning a cloud data center with high availability level and the cheapest acquisition cost of equipment is a challenge for data center operators. Without a proper equipment selection to achieve the best cloud service availability may affect provider's business or even worse compromise its reputation. We addressed this problem by using SPN models of different data center architectures based on the TIA-942 standard proposed by [9], [10], [11], [12]. Then, we proposed, compared and validated surrogate models to handle the complexity of the SPN models, to make it feasible to be applied in optimization algorithms. Lastly, we applied the PSO, DE, and GA algorithms to find the optimal solution and compared the availability level found by them according to different budget constraints. Results regarding the comparison of PSO, DE, and GA algorithms show that DE obtained the best results (highest availability for different budget constraints) followed by GA, and PSO.

As future works, we plan to apply other optimization algorithms and compare them; and we also intend to evaluate other strategies for making the surrogate model, as machine learning algorithms.

References

[1] P. T. Endo, G. E. Gonçalves, D. Rosendo, D. Gomes, G. L. Santos, A. L. C. Moreira, J. Kelner, D. Sadok, and M. Mahloo, "Highly available clouds: System modeling, evaluations, and open challenges," in *Research Advances in Cloud Computing*, pp. 21–53, Springer, 2017.

[2] L. A. Barroso, J. Clidaras, and U. Hölzle, "The datacenter as a computer: An introduction to the design of warehouse-scale machines," *Synthesis lectures on computer architecture*, vol. 8, no. 3, pp. 1–154, 2013.

[3] P. T. Endo, G. L. Santos, D. Rosendo, D. M. Gomes, A. Moreira, J. Kelner, D. Sadok, G. E. Gonçalves, and M. Mahloo, "Minimizing and managing cloud failures," *Computer*, vol. 50, no. 11, pp. 86–90, 2017.

[4] M. N. Cheraghlou, A. Khadem-Zadeh, and M. Haghparast, "A survey of fault tolerance architecture in cloud computing," *Journal of Network and Computer Applications*, vol. 61, pp. 81–92, 2016.

[5] G. Wu, R. Mallipeddi, and P. Suganthan, "Problem definitions and evaluation criteria for the cec 2017 competition on constrained real-parameter optimization," *Technical Report*, 2016.

[6] K.-L. Du and M. Swamy, "Particle swarm optimization," in *Search and Optimization by Metaheuristics*, pp. 153–173, Springer, 2016.

[7] U. Chakraborty, *Advances in differential evolution*, vol. 143. Springer, 2008.

[8] D. Whitley, "An executable model of a simple genetic algorithm," *Foundations of genetic algorithms*, vol. 2, no. 1519, pp. 45–62, 2014.

[9] D. M. Gomes, P. T. Endo, G. Gonçalves, D. Rosendo, G. L. Santos, J. Kelner, D. Sadok, and M. Mahloo, "Evaluating the cooling subsystem availability on a cloud data center," in *Computers and Communications (ISCC), 2017 IEEE Symposium on*, pp. 736–741, IEEE, 2017.

[10] G. L. Santos, P. T. Endo, G. Gonçalves, D. Rosendo, D. Gomes, J. Kelner, D. Sadok, and M. Mahloo, "Analyzing the it subsystem failure impact on availability of cloud services," in *Computers and Communications (ISCC), 2017 IEEE Symposium on*, pp. 717–723, IEEE, 2017.

[11] D. Rosendo, G. Leoni, D. Gomes, A. Moreira, G. Gonçalves, P. Endo, J. Kelner, D. Sadok, and M. Mahloo, "How to improve cloud services availability? investigating the impact of power and it subsystems failures," in *Proceedings of the 51st Hawaii International Conference on System Sciences*, 2018.

[12] D. Rosendo, P. T. Endo, G. L. Santos, D. M. Gomes, G. Goncalves, A. Moreira, J. Kelner, D. Sadok, and M. Mahloo, "Modeling and analyzing power system failures on cloud services," in *2017 13th International Conference on Network and Service Management (CNSM)*, pp. 1–7, IEEE, 2017.

[13] M. Asher, B. Croke, A. Jakeman, and L. Peeters, "A review of surrogate models and their application to groundwater modeling," *Water Resources Research*, vol. 51, no. 8, pp. 5957–5973, 2015.

[14] T. Evans, "The different technologies for cooling data centers." `http://www.apc.com/salestools/VAVR-5UDTU5/VAVR-5UDTU5_R2_EN.pdf`, 2012. Last access: February, 2017.

[15] E. Andrade, B. Nogueira, R. Matos, G. Callou, and P. Maciel, "Availability modeling and analysis of a disaster-recovery-as-a-service solution," *Computing*, pp. 1–26, 2017.

[16] A. Moreira, D. Rosendo, D. Gomes, G. Santos, L. Silva, C. Cani, J. Kelner, D. Sadok, G. Gonçalves, A. Mehta, M. Wildeman, and P. Endo, "Dcav: a software system to evaluate next-generation cloud datacenter availability through a friendly graphical interface," *Software: Practice and Experience*, p. to appear, 2019.

[17] G.-G. Wang, A. Hossein Gandomi, X.-S. Yang, and A. Hossein Alavi, "A novel improved accelerated particle swarm optimization algorithm for global numerical optimization," *Engineering Computations*, vol. 31, no. 7, pp. 1198–1220, 2014.

[18] R. Rao, "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 7, no. 1, pp. 19–34, 2016.

[19] V. Roberge, M. Tarbouchi, and G. Labonté, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 132–141, 2013.

[20] L. Tang, Y. Zhao, and J. Liu, "An improved differential evolution algorithm for practical dynamic scheduling in steelmaking-continuous casting production," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 2, pp. 209–225, 2014.

[21] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution–an updated survey," *Swarm and Evolutionary Computation*, vol. 27, pp. 1–30, 2016.

[22] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (abc) algorithm," *Applied soft computing*, vol. 8, no. 1, pp. 687–697, 2008.

[23] P. Ghamisi and J. A. Benediktsson, "Feature selection based on hybridization of genetic algorithm and particle swarm optimization," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 2, pp. 309–313, 2015.

[24] M. Malhotra and K. S. Trivedi, "Power-hierarchy of dependability-model types," *IEEE Transactions on Reliability*, vol. 43, no. 3, pp. 493–502, 1994.

[25] L. P. Swiler, P. D. Hough, P. Qian, X. Xu, C. Storlie, and H. Lee, "Surrogate models for mixed discrete-continuous variables," in *Constraint Programming and Decision Making*, pp. 181–202, Springer, 2014.

[26] P. Kersaudy, B. Sudret, N. Varsier, O. Picon, and J. Wiart, "A new surrogate modeling technique combining kriging and polynomial chaos expansions–application to uncertainty analysis in computational dosimetry," *Journal of Computational Physics*, vol. 286, pp. 103–117, 2015.

[27] K. Elsayed and C. Lacor, "Robust parameter design optimization using kriging, rbf and rbfnn with gradient-based and evolutionary optimization techniques," *Applied Mathematics and Computation*, vol. 236, pp. 325–344, 2014.

[28] Z. Zhou, Y. S. Ong, P. B. Nair, A. J. Keane, and K. Y. Lum, "Combining global and local surrogate models to accelerate evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 1, pp. 66–76, 2007.

[29] C. Lataniotis, S. Marelli, and B. Sudret, "Uqlab user manual–kriging (gaussian process modelling)," *Report UQLab-V0*, pp. 9–105, 2015.

[30] M. A. Oliver and R. Webster, "Kriging: a method of interpolation for geographical information systems," *International Journal of Geographical Information System*, vol. 4, no. 3, pp. 313–332, 1990.

[31] A. Florian, "An efficient sampling scheme: updated latin hypercube sampling," *Probabilistic engineering mechanics*, vol. 7, no. 2, pp. 123–130, 1992.

[32] M. Sadeghi, M. Pashaie, and A. Jafarian, "Rbf neural networks based on bfgs optimization method for solving integral equations," *Adv. Appl. Math. Biosci*, vol. 7, no. 1, pp. 1–22, 2016.

[33] X.-S. Yang, *Nature-inspired optimization algorithms*. Elsevier, 2014.

[34] M. E. H. Pedersen, "Good parameters for particle swarm optimization," *Hvass Lab., Copenhagen, Denmark, Tech. Rep. HL1001*, 2010.

[35] M. E. H. Pedersen, "Good parameters for differential evolution," *Magnus Erik Hvass Pedersen*, vol. 49, 2010.

[36] U. Kumar, "System maintenance: Trends in management and technology," in *Handbook of Performability Engineering*, pp. 773–787, Springer, 2008.

[37] K. McCarthy and V. Avelar, "Comparing ups system design configurations," *APC white paper 75, Schneider electric Data center science center*, 2005.

[38] G. Callou, P. Maciel, D. Tutsch, J. Ferreira, J. Araújo, and R. Souza, "Estimating sustainability impact of high dependable data centers: a comparative study between brazilian and us energy mixes," *Computing*, vol. 95, no. 12, pp. 1137–1170, 2013.

[39] A. P. Guimaraes, P. Maciel, and R. MATLAs JR, "Design of it infrastructures of data centers: An approach based on business and technical metrics," *Quantitative Assessments of Distributed Systems: Methodologies and Techniques*, p. 265, 2015.

[40] S. Shetty, "Determining the availability and reliability of storage configurations," *Dell Power Solutions*, 2002.

[41] B. Schroeder and G. A. Gibson, "Disk failures in the real world: What does an mttf of 1, 000, 000 hours mean to you?," in *FAST*, vol. 7, pp. 1–16, 2007.

[42] C. Heising *et al.*, "Ieee recommended practice for the design of reliable industrial and commercial power systems," *IEEE Inc., New York*, 2007.

[43] F. Nwanganga, M. Saebi, G. Madey, and N. Chawla, "A minimum-cost flow model for workload optimization on cloud infrastructure," in *Cloud Computing (CLOUD), 2017 IEEE 10th International Conference on*, pp. 480–487, IEEE, 2017.

[44] Z. Zhou, J. H. Abawajy, F. Li, Z. Hu, M. Chowdhury, A. Alelaiwi, and K. Li, "Fine-grained energy consumption model of servers based on task characteristics in cloud data center," *IEEE access*, 2017.

[45] K. Chatziprimou, K. Lano, and S. Zschaler, "Surrogate-assisted online optimisation of cloud iaas configurations," in *Cloud Computing Technology and Science (CloudCom)*, pp. 138–145, IEEE, 2014.

[46] F. Sheikholeslami and N. J. Navimipour, "Service allocation in the cloud environments using multi-objective particle swarm optimization algorithm based on crowding distance," *Swarm and Evolutionary Computation*, vol. 35, pp. 53–64, 2017.