Multi-core synthesis and maximum satisfiability applied to optimal sizing of solar photovoltaic systems

SUBMISSION DATE / POSTED DATE

23-04-2021 / 28-04-2021

CITATION

Trindade, Alessandro; Galvão, Edilson; Cordeiro, Lucas (2021): Multi-core synthesis and maximum satisfiability applied to optimal sizing of solar photovoltaic systems. TechRxiv. Preprint. https://doi.org/10.36227/techrxiv.14474235.v1

DOI

10.36227/techrxiv.14474235.v1

Multi-core synthesis and maximum satisfiability applied to optimal sizing of solar photovoltaic systems

Edilson Galvão, Alessandro Trindade and Lucas Cordeiro

Abstract—Annual global energy consumption growth is around 1.3% with forecasts until 2040. Photovoltaic systems became a suitable alternative to nuclear and fossil energy generation. In order to support this technology's dissemination, we develop and evaluate an automated formal synthesis approach that assists in decision-making for off-grid systems. Our proposed approach, called PVz, is based on a variant of the counterexample-guided inductive synthesis; it has a multi-core feature, which can obtain the optimal sizing of photovoltaic systems focusing on Life Cycle Cost analysis. Given the electrical needs of a home, we seek a set of electrical equipment with the best possible combination of devices that meet the specified requirements. We calculate all costs related to maintenance over 20 years. The results presented are based on seven case studies; some of them are real ones from the Amazon region in Brazil. The same case studies were solved by a commercial optimization tool. Our technique and the commercial tool results were validated with popular simulation software to perform a fair comparison. Furthermore, we analyze some topics such as run-time, optimal solution, and configuration of the resulting systems. We claim that our technique is advantageous compared to the existing approaches in the literature.

Index Terms—Formal synthesis, software verification, model checking, solar photovoltaic systems.

1 Introduction

The recent studies about global energy indicate that 789 million people have no access to electricity, which is 10% of the world population [1]. From 2010 until 2018, the effort to reduce the number of people without electricity access increased, and the result is positive. Quantitatively the result was a decrease from 1.2 billion to 0.84 billion people without electrical energy. Out of this total, renewable energy solutions are responsible for 136 million people receiving basic energy service [1]. Unfortunately, lack of access to clean and affordable energy is considered a core dimension of poverty [2]; this has a direct impact on the low Human Development Index (HDI) of different localities [3]. It follows that increased access to energy allows economic growth and poverty alleviation [4].

To provide electricity for all, decentralized systems led by solar photovoltaic (PV) in off-grid and mini-grid systems will be the lowest-cost solution for three-quarters of the connections needed [2]. Thus, this can be ratified by analyzing the result of renewable energy reported by the global bank, where 17.3% of global energy is based on wind and solar energy [1].

Changes to improve the electrical energy uses, the more precise the sizing for electrical grids or stand-alone systems, the more meaningful the use of renewable energies. For this purpose, some software is available in the market; a part of them is created for general-purpose as MATLAB [5] and others for specific electrification studies like RETScreen, and HOMER [6], [7]. However, the industry demands the design solution to be the optimum, considering equipment manufacturers and models available on the market and not just minimum or maximum values of current or power for the optimized items [8], [9]. We need to evaluate the electrical compatibility among the equipment, which can only be achieved with specialized PV optimization software. Therefore, the optimal solution is the lowest cost from a list of equipment that meets the house's electrical demands.

Given the above, the HOMER tool manages to deliver an optimal solution in a smaller scope, where only batteries and solar panels are optimized, while all other modules are simulated. In contrast, the technique described in this work, called PVz, is capable of optimizing, in addition to solar panels and batteries, electrical inverters, and controllers. The clear difference between the optimized electrical devices ensures greater correctness once both techniques are compared, thus favoring more extensive design-space coverage.

Here, we have developed a variant of the counterexample guided inductive synthesis (CEGIS) [10] for synthesizing optimal sizing of stand-alone PV systems using commercial equipment data. If the user gives a correctness specification σ, our technique uses that as a starting point and then iteratively produces a sequence of candidate solutions that satisfy σ, related to power reliability.

We used a solver called Z3 [11]. Internally, the Z3 tool contains a module to include optimization objectives. This module is named νZ and integrates state-of-the-art algorithms for optimization and extra tools to solve linear

- *E. Galvão is with MSc student in the Post-graduate Program in Electrical Engineering, Manaus, Brazil, e-mail: esj.galvao@gmail.com.*
- *A. Trindade is with the Department of Electricity, Federal University of Amazonas, Manaus, Brazil, e-mail: alessandrotrindade@ufam.edu.br.*
- *L. Cordeiro is with School of Computer Science, The University of Manchester, UK, e-mail: lucas.cordeiro@manchester.ac.uk.*

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.
Manuscript received March XX, 2021; revised XXXXX 11, 2021.

restrictions problems. In particular, the νZ features match our problem and will help us to optimize PV systems [11]. In each iteration, we synthesize the sizing of stand-alone PV systems, but that may not achieve the lowest cost. Thus, the candidate solution passes through an SMT solver to reduce the number of possible states. Our candidate solution provides a lower bound, which serves as the minimum cost of reference to help our engine restrict verification. Note that each iteration can be lower or higher than the current reference, once found a value lower than the current reference, and it respects the user constraints. The reference is updated globally. If the verification by the SMT solver step does fail, it means there is an optimal solution. Thus, all content produced is saved as a counterexample with an optimal sizing that meets both power reliability and system cost. If the verification step does not fail, the content produced is ignored. Our novelty relies on a practical approach to pursuing the optimal solution of PV systems using formal methods.

Research in the fields of renewable energy and photovoltaic systems, date from the mid-90s, and bring different approaches to find an optimal solution for the design of electrical systems as can be seen in [12] and [9]. However, the amount of work related to the formal modeling of photovoltaic systems is still low. Consequently, it is even more challenging to find studies based on state-of-the-art solvers as ESBMC [13], CPAchecker [14] and Z3 [11]. The starting point for this work [15] [16] [17] are studies that derive from the year 2019 and converge to analyze the optimization problem of photovoltaic systems using SMT solvers and commercial tools such as HOMER and PVsyst.

This paper makes the following original contributions. First, it is a radical improvement in the base paper's performance describing formal synthesis for stand-alone PV systems application [8]. Second, the technique described here supports data processing parallelization since it relies on a multi-core processor. Third, the state space coverage was increased compared to the base article [15]. It was possible to cover and compute all the spaces of the combinations of electrical equipment. Fourth, experimental results with seven case studies show that the formal synthesis approach qualitatively outperforms an existing state-of-the-art optimization tool. Lastly, the results are validated with accurate commercial design software called PVsyst.

2 Background

Fig. 1 illustrates how to obtain the optimal sizing of a stand-alone PV system using two different modes. The first one is the traditional model called ***traditional sizing***, which includes manual verification by computing the electrical equipment capacity and installation for validation. Concerning the first mode, there are still commercial tools as MATLAB [5] and HOMER [6], [7], which simulate the real electrical system. The second includes ***Automated Synthesis*** that encompasses verification engines. This option proves to be viable since it generates more complex and accurate data than the traditional way described previously. Both modes described here use the same input and theoretically produce the same output; it is noteworthy that variables such as electrification, support for different categories of equipment, and different electrical sizing forms impact the expected output.

For **input** data, we can consider weather data, price information about each piece of equipment, design requirements, load curve, power demand, and design assumptions. This information is converted into a matrix of data and exposed as global content to the software.

For the **output** data and a more straightforward reading of this work, we will assume from now on that every benchmark that contains a valid answer will be called **satisfiable (SAT)**. By contrast, when there is no solution, we will call it **unsatisfiable (UNSAT)**. If there is a set of renewable energy equipment, which can be combined (price and system hardware composition), the result is SAT.

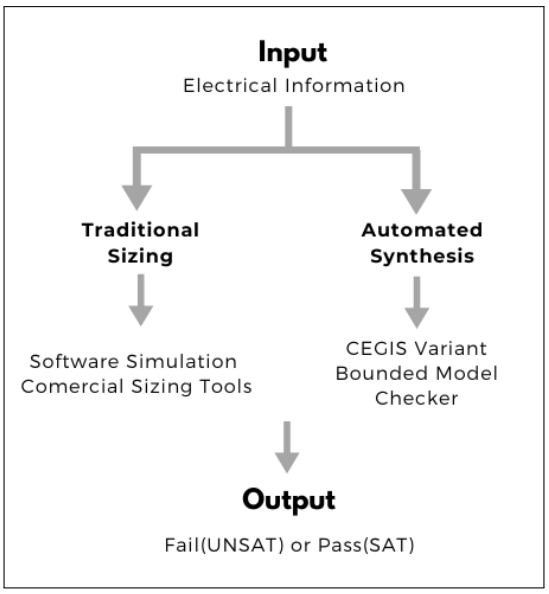


Fig. 1. Comparative image of the traditional method versus the proposed method.

The automated synthesis technique is mathematical reasoning about a formal model, where the SAT result is a counterexample. The counterexample is intended to record changes in the values of the program variables during program execution. In this work, since the system's answer is SAT, the counterexample will show the status of all variables when the solution was found. Thus, the answer can guide the software or client to choose the best combination of electrical machines. The traditional way does not offer the quantity of necessary data about the system and internal modes [5], [6], [7]. Another notable topic that needs to highlight is each model's coverage; models that use Automated Synthesis (via BMC engines) can ensure complete coverage over the entire state-space, instead of locals states as a traditional model.

2.1 Program Synthesis

The basic idea of program synthesis is to automatically construct a P program that satisfies a correctness specification σ. In particular, program synthesis is automatically performed by engines that use a correctness specification σ, as a starting point and then incrementally produce a sequence of candidate solutions that partially satisfy σ [18].

As a result, a given candidate program p is iteratively refined to match σ more closely. Figure 2 illustrates the underlying architecture.

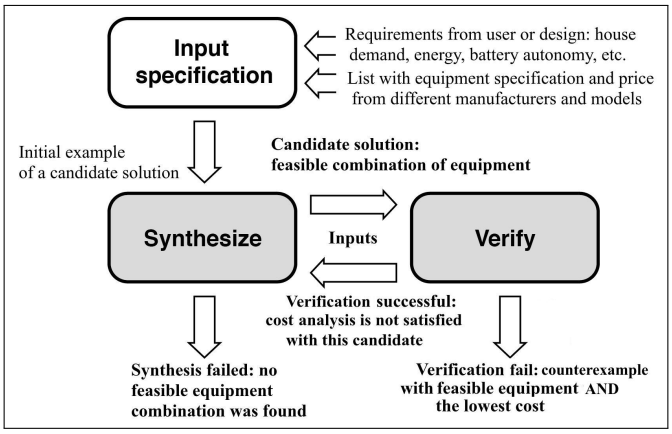


Fig. 2. CEGIS in PV system sizing.

The correctness specification σ provided to our synthesizer is of the form $\exists\vec{F}.\forall\vec{x}.\sigma(\vec{x},\vec{F})$, where $\vec{F}$ ranges over functions, $\vec{x}$ ranges over ground terms, and σ is a quantifier-free (QF) formula typically supported by SMT solvers. The ground terms are interpreted over some finite domain $\mathcal{D}$, where $\mathcal{D}$ can be encoded using the SMT's bit-vectors part. Our specification includes house demand, energy, and battery autonomy; we also provide equipment specifications and prices from different manufacturers and models.

In Figure 2, the phases SYNTHESIZE and VERIFY interact via a finite set of test vectors INPUTS, which is incrementally updated. Given the correctness specification σ, the SYNTHESIZE procedure tries to find an existential witness $\vec{F}$ satisfying the specification $\sigma(\vec{x},\vec{F})$, for all $\vec{x}$ in INPUTS (as opposed to all $\vec{x} \in \mathcal{D}$). If SYNTHESIZE succeeds in finding a witness $\vec{F}$, the latter is a candidate solution to the full synthesis formula, which is passed to VERIFY to check whether it is a proper solution (*i.e.*, $\vec{F}$ satisfies the specification $\sigma(\vec{x},\vec{F})$ for all $\vec{x} \in \mathcal{D}$). If this is the case, then the algorithm terminates.

One may notice that each iteration of the traditional CEGIS loop adds a new input to the finite set $INPUTS$, which is then used for synthesis. Given that the full set of inputs $\mathcal{D}$ is finite because we use bit-vector expressions, the refinement loop can only iterate over a finite number of times. However, SYNTHESIZE may conclude that no candidate solution obeying σ for the finite set $INPUTS$ exists.

In our CEGIS variant, there exist four differences related to the traditional one: (1) there exists no test vector, and every candidate is generated in the SYNTHESIZE phase and sent to the VERIFY phase; (2) if the VERIFY phase is unsuccessful, then a new candidate is generated by SYNTHESIZE and (3) the lower bound of the VERIFY phase is incremented to search for the lowest cost; as a result, (4) there exists no refinement from the VERIFY phase back to the SYNTHESIZE phase. In particular, a new counterexample is not added to the INPUT set since a failure during the VERIFY phase will only discard a given candidate, which could be feasible in the next iteration with a new lower bound.

In summary, our proposal is a technique based on CEGIS, which aims to synthesize the optimal solution of a PV system; therefore, our technique addresses an optimization problem.

2.2 Sizing Stand-alone Solar PV Systems

A PV system is illustrated in Fig.3. It employs the PV generator (*panel or an array*), a semiconductor device that can convert solar energy into DC electricity. We hold *batteries*, where power can be stored and used for night hours or rainy days. The use of batteries as a storage form implies a *charge controller* [19]. The PV arrays produce DC, and therefore when the PV system contains an AC load, a DC/AC conversion is required (*inverter*). The *AC load* dictates the AC electrical load's behavior from the house that the system will feed.

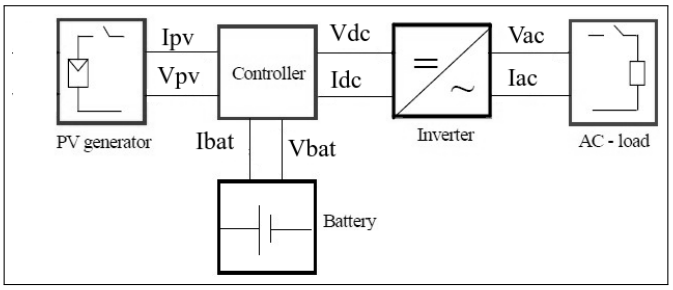


Fig. 3. Block diagram for a typical stand-alone PV system [19].

The sizing check stage can ensure that the system meets the standard project steps related to the critical period method (worst month) for solar energy system sizing [20]. It adopts an MPPT (Maximum Power Point Tracking) charge controller, which is the most common in use.

Since this paper's audience is targeted to be from the software verification area, we decided to use a higher-level explanation about the PV sizing. The sizing process involves eighteen equations related to the electrical engineering area, which is detailed online.[1] Fig. 4 illustrates the overview of the steps that must be taken to size a stand-alone PV system.

On the left side of Fig. 4, we describe the needed **inputs** to size a PV system. There are requirements from the house to be electrified; in particular, we have the design assumptions, weather information from the targeted local of the PV system deployment, and a possible initial list of equipment to cover all the items listed Fig. 3. Concerning the equipment list, the designer can use a few pieces of equipment or a vast one. We can also decide not to adopt the commercial equipment list. However, the result is a PV sizing of specific power, current, or voltage values, usually just close to original equipment, found in the market. This possibility has the drawback of possible incompatibility among equipment when the real one is bought and deployed.

There exist specific steps that aim the calculation of some variables and others related to the electrical compatibility validation among equipment items, both enumerated from **i** to **xiv**, as illustrated in Fig. 4; those represent different

1. https://cutt.ly/Dz1Ua6Q

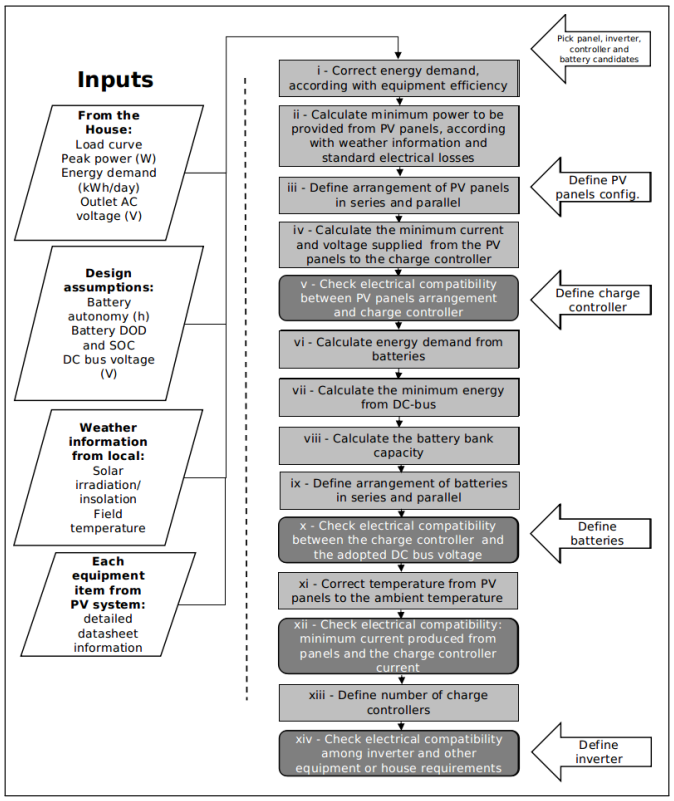


Fig. 4. High level description of stand-alone PV system sizing process.

shades of gray of the rectangle boxes. The start point is usually a candidate list of PV panels, charge controller, battery, and inverter, as indicated at the top of the flowchart. The arrows on the right side show a point where some specific item is validated. The diagram does not show the returning location. However, if the candidate item is not compatible with others or does not meet some requirements, it must be changed to follow the indicated flow. The last rectangle box checks the inverter electrical compatibility with the DC-bus voltage, with the outlet's required AC voltage. The inverter specified power must be lower than the charge controller power to avoid burning by overcharge. At the end of the flowchart, all the items are defined, and the PV sizing is finished.

These equations model the PV system's continuous-time behavior; they produce real numbers except for the batteries and panels. Real numbers must be converted into integer ones, considering the minimum or maximum according to each equation. The verification and simulation tools need to handle non-linear real arithmetic to produce the correct result. Our mathematical model uses floating-point arithmetic. It is just an approximation of the real numbers. However, in this work, we are not concerned with calculating the rounding error, which is negligible when considering the size of the physical quantities and the variables adopted [21].

3 Synthesizing Optimal Sizing of Stand-alone Solar Photovoltaic Systems

The best compromise between two objectives makes the optimal sizing of PV systems: *power reliability* and *system cost* [22]. This study will rely on the critical period solar energy method [20], as described in Section 2.2. Our study will use an adapted Life Cycle Cost (LCC) analysis, where the acquisition cost of every item of equipment is considered, plus the installation cost, the operational and maintenance costs [22]; theses costs are represented by:

$$LCC = EC + EM \tag{1}$$

EC denotes the costs of the following equipment: C_{PV} is the PV array cost, C_{bat} is the initial cost of batteries, $C_{charger}$ is the cost of the charger, C_{inv} is the inverter cost.

$$EC = C_{PV} + C_{bat} + C_{charger} + C_{inv} \tag{2}$$

EC denotes other costs related to the maintenance and proper functioning of the electrical system: $C_{installation}$ is the installation cost, C_{batrep} is battery replacement cost at current prices, and $C_{PWO\&M}$ is operation and maintenance costs at current rates.

$$EM = C_{installation} + C_{batrep} + C_{PWO\&M} \tag{3}$$

In this study, we will use a $C_{installation}$ equivalent to 5% of total equipment cost and a $C_{PWO\&M}$ equal to U$ $289.64/year$, according to Amazon State literature data [23]; and an LCC lifetime analysis of 20 years.

In this section, we will describe our algorithms and how model checking can be used as a back-end verification engine [8]. Algorithm 1 describes how to save and expose the correct solutions and the optimal solution, and Algorithm 2 is responsible for verifying the mathematical assertion over electrical equipment.

Algorithm 1: Find by the optimal solution

```
Result: returns a feasible sizing of PV system with
        the lowest cost
1  Initialize arrays and variables;
2  Create Satisfable List called NodeSatList;
3  Create AllAvailableNodes list;
4  Create maximumCost as a long variable;
5  forall AllAvailableNodes node do
6      isLowestNode = solve(node);
7      if isLowestNode = true then
8          lock(this);
9          maximumCost = node.cost;
10         NodeSatList.add(node);
11         unlock(this);
12     end
13 end
14 return NodeSatList.OrderBy(x.Cost).FirstOrDefault()
```

Algorithm 1 receives as input the data described in Fig. 1. This content is represented in matrices and global variables. The first line includes the manufacturer's data, prices of PV panels, batteries, charge controllers, and inverters. Lines 2, 3, and 4 represent the state-space, the nodes that satisfy the equations, and the lowest global cost.

AllAvailableNodes is the data set representing the total sample of states to be explored; its purpose is to ensure 100% coverage of equipment combinations. A node represents a feasible arrangement or not of proposed electrical equipment for the test case; that is, among all the

solutions and ways of organizing the photovoltaic system under study, a node corresponds to one possible solution. We have an object-oriented class that stores all relevant information. For example, electrical equipment that will be combined, the solar radiation level in the region, best global equipment cost, lowest local equipment cost, consumption, and electrical details of each test case) so that the solver can process the set of equipment and validate it.

The representation of the problem through nodes decreases the complexity of processing data in physical processor architectures that accept multi-threads. Therefore, the parallel loop is used to process each node separately. The function *solve* described in Algorithm 2 returns true when the node is a possible solution, and the cost is lower than the global maximum cost. Therefore, to ensure the proper storage of results processed by many threads simultaneously, the lock function was added. After the internal loop ends, the last action returns the optimal solution found in the list that contains all possible solutions.

Algorithm 2: Verify if the node is satisfactory for restrictions and equipment.

Result: Returns the possible cost of the F combination of equipment.

1 Initialize internal arrays and variables;
2 **Function** `Solve(`F`)`**:**
3 Declare non-deterministic variables to select PV Panel, Controller, Battery, and Inverter from list;
4 Calculate Steps i and ii of Fig. 4;
5 Define PV panels arrangement: Step iii of Fig. 4;
6 Calculate Step iv of Fig. 4;
7 Enforce electrical compatibility in Step v of Fig. 4 with statement **assume**;
8 Calculate Steps vi to viii of Fig. 4;
9 Define battery arrangement according Step ix of Fig. 4;
10 Enforce electrical compatibility in Step x of Fig. 4 with statement **assume**;
11 Correct variables to ambient temperature: Step xi of Fig. 4;
12 Enforce electrical compatibility in Step xii of Fig. 4 with statement **assume**;
13 Define number of charge controllers: Step xiii of Fig. 4;
14 Enforce electrical compatibilities in Step xiv of Fig. 4 with statement **assume** and define the inverter;
15 Non-deterministic variables hold feasible equipment and cost;
16 $F_{obj} \leftarrow N_{TP} * Panel_{Cost} + N_{TB} * Battery_{Cost} + Controller_{Cost} + Inverter_{Cost} + Installation_{Cost} + batrep_{Cost} + PWO\&M_{Cost}$ **return** isSatisfiable;
17 **End Function**;

To choose the best option, the system uses as input a list of **seventy** equipment from **twelve** different manufacturers, where each technical information required was found in a datasheet provided by the equipment factory. To calculate Eq. (1) value, it was necessary to convert the amounts to US dollars based on the exchange rate of the day. To simplify understanding about this type of data, a report has been created that condensates general information required to perform tests over this tool, and it is available online.[2]

Algorithm 1 retrieves the information described above based on arguments provided by the *solve* function and creates an internal context. This context is responsible for verifying whether all equations are described in Algorithm 2, specified in lines 06, 08, 11, 15 are satisfiable. Besides, the *assume* method is called to ensure that the restrictions described in lines 07, 10, 12 and 14 are met.

This algorithm contains three essential points. The **first** point is the set of equations that correspond to a valid combination of electrical components. These equations can be found in lines 04, 05, 06, 08, 09, 11, 13, and 15. The **second** point refers to all four *assume* methods, which are necessary since they serve as a limit or barrier of acceptable values to pieces of equipment. These restrictions can be found in lines 07, 10, 12, and 14. The **third** point, the return system that can be SAT (i.e., a solution was found) or UNSAT (i.e., no solution was found).

The algorithm 2 always returns two different states. SAT when the electrical equipment combinations and the general cost are safe to become a possible solution. UNSAT once the program finishes without finding a solution, indicating that it could not combine the specific equipment items to create a feasible solution. In some scenarios, we can expect a memory overflow or excessive time (timeout) resulting in a system with no concrete result; it is most common in hardware with few gigabytes of memory. The main challenge for Algorithm 2 is to find a feasible candidate solution for the constraints and user requirements.

In summary, we use four non-deterministic variables to index four matrices with complete datasheet[2] information from an equipment item. We have four variables and four matrices: one to PV panels, one to batteries, one to the inverter, and one to the charge controller. Those non-deterministic variables are used during the search for the feasible solution and controlled by the statements assume. Note that the process described here is completely automated to ensure that the solution is sound. The verification engines transform the Algorithm 2 into the Boolean expressions that are passed to the solver to verify ($C \wedge \neg P$), as described online.[3]

4 Results and Discussion

4.1 Objectives and Setup

Our evaluation aims to answer three experimental goals:

- EG1 **(soundness)** Does our automated synthesis approach provide correct results?
- EG2 **(performance)** How do the software verifiers compare to each other for synthesizing PV systems?
- EG3 **(state-of-the-art)** how does our formal synthesis tool compare to a specialized simulation tool?

All experiments were conducted on an otherwise idle Intel Xeon CPU E5-4617 (6-cores and 6 Threads) with 2.90 GHz and 64 GB RAM, running Ubuntu 18.04 LTS 64-bits.

2. https://cutt.ly/Vz1Uw84
3. https://cutt.ly/gz1Y159

For HOMER Pro, we have used an Intel Core i5-4210 (4-cores) with 1.7 GHz and 4 GB RAM running Windows 10. The ideal scenario would be to use the same hardware configuration for the experiments. However, this setup has an impact on performance, which is less favorable to HOMER Pro. PVsyst used the same configuration as HOMER Pro. We perform the experiments with a predefined *timeout* of $200,000$ seconds (55.5 hours).

We evaluated three state-of-the-art verifiers, Z3 [4] version 4.8.9 with νZ as optimization engine, ESBMC[5] v6.0.0 [13] with the Boolector 3.0.1 solver [24], and CPAchecker[6] v2.0 [14] with MathSAT 5.6.5 [25], were used as verification engines to compare the proposed approach effectiveness and efficiency.

4.2 Description of Benchmarks

The proposed synthesis approach was evaluated in seven stand-alone PV systems. These benchmarks were defined based on the usual electrical load found in riverside communities in the Amazonas State, Brazil [16], [23], except for case 7, which was idealized to support a few lamps and a 12k BTUs air-conditioner solution. Here, we report each case study as a 4-tuple {*power peak (W); power surge (W); energy consumption (Wh/day); battery autonomy (hours)*} as follows: **1:** {342; 342; 3,900; 48}; **2:** {814; 980; 4,880; 48}; **3:** {815; 980; 4,880; 12}; **4:** {253; 722; 3,600; 48}; **5:** {263; 732; 2,500; 48}; **6:** {322; 896; 4,300; 48}; **7:** {1,586; 2,900; 14,000; 48}. This 4-tuple represents the Algorithm 2 inputs. For all cases, an estimated load curve (kWh) was defined based on the electronics consumers in each house. Our synthesis algorithm was fed with data and costs of seventy equipment items from twelve different manufacturers of PV systems.

4.3 Solvers and State Space

Classical software verifiers such as ESBMC and CPAchecker will explore the state space by searching the proposed problem's solution. Each verified state demands computational cost; proportionally, the greater the number of states to be processed, the longer the problem is resolved, and the greater the computational power is required. The Z3 tool has an API that works specifically for solving optimization problems. This tool substantially reduces the computational capacity required and converges to this work's objective described in Section 4.1 since the vZ solver is focused on solving optimization problems and providing optimization of linear problems on SMT formulas.

Often, many states will cause classical SMT solvers to demand more time than is acceptable [26]. As described in 4.1, this would override the EG01 objective since it would not be possible to verify that all tools compared here produce correct answers. We will then use two approaches, called **Reduced** and **Expanded** respectively, which will guarantee the demonstration of correctness and the robustness of the proposed system. The first reduces the state-space by ensuring that all solvers described in the previous paragraph will have an answer to the problem promptly. The second, called **Expanded**, guarantees the robustness of the proposed technique. We will significantly increase the space of states and verify the performance differences compared to the market programs. It is important to note that in both **Reduced** and **Expanded** approaches, the combination of the photovoltaic system will be the same when analyzing the results of the tools.

4.4 Simulation Tools and Assumptions

Concerning the off-the-shelf optimization/simulation tools, only HOMER Pro performs an off-grid system with battery backup analysis and includes economical analysis [6], [7]. Here we used HOMER Pro version 3.13.1 as a state-of-the-art optimization tool for comparison purposes. In particular, HOMER Pro has the following characteristics: (a) it is available for MS-Windows only; its annual standard subscription costs US$ $1,500.00$ using Expert Package [27]; (b) it has two optimization algorithms: one algorithm simulates all of the feasible system configurations defined by the search space, and additionally, a proprietary derivative-free algorithm to search for the least-costly system; (c) it does not have LCC cost in its reports, only Net Present Cost (NPC); however, we can obtain LCC from NPC; (d) the optimization analysis defines a load curve and temperature according to data collected from online databases.

However, to allow a correct comparison, the curve load and the temperature were defined the same as our synthesis approach; (e) it does not have a charge controller. During the tests, we have chosen the "load-following" option, which produces enough power to meet the demand [27] and (usually) presents a non-overestimated solution; (f) it was assumed 95% availability of the PV system. By definition, "availability" is the percentage of time at which a power system can feed the load requirements [28]. For an ordinary house electrical load, 95% is considered acceptable; (g) it was assumed a string of two batteries to match the voltage of the 24 V DC system, which was used for our automated synthesis tool; (h) it was included a generic flat-plate PV of 1 kW and generic lead-acid batteries of 1 kW (83.4 Ah capacity). During runtime, HOMER decides the size in kW of each one based on feasibility and lower cost.

To validate and compare the optimal sizing solution produced by our approach and by HOMER Pro, we use a simulation tool, called PVsyst version 6.86 [29], with plenty of commercial equipment in its database. We have considered a comparison for an entire year's weather data of simulation to ensure that the proposed sizing meets the electrification requirements. PVsyst is a PC software package developed by a Swiss company used for the study, sizing, simulation, and data analysis of solar PV systems. PVsyst contains design, sizing, 3D shading scene, simulation, grid, and off-grid features. It uses comprehensive irradiation data from Meteonorm,[7] and aging analysis [30]. However, it does not perform optimization; therefore, PVsyst needs the system sized to validate it. Furthermore, PVsyst does not have commercial inverter equipment and, as a result, does not consider surge power demand as the ones produced by

4. Command-line: $ dotnet run TC-ID

5. Command-line: $ esbmc filename.c --incremental-bmc --boolector

6. Command-line: $ scripts/cpa.sh -heap 64000m -config config/bmc-incremental.properties -spec config/specification/sv-comp-reachability.spc file.c

7. https://meteonorm.com/en/

air conditioners and refrigerators for a few seconds. PVsyst is commercial software with a 30-day test possibility and runs only in MS Windows.

4.5 Results

Table 1 shows the result of both approaches described in Section 4.3. This table is split into four important sets of columns, read from left to right: Column 01 describes all benchmarks with their respective specifications. Columns 02, 03, and 04 refer to the reduced approach that proves the system's correctness. Columns 05, 06, and 07 refer to the expanded approach, proving the commercial analysis tool's robustness. Finally, the last column refers to the simulation of the photovoltaic system using HOMER Pro.

Reduced approach: Traditional software verifiers, e.g., ESBMC [13] and CPAchecker [14], face difficulties in solving optimization problems. However, they can be solid allies for detecting the optimal solution. The idea behind this approach is to reduce state-space exploration. The reduction in the state space was due to the reduction in the number of electrical equipment combinations as described in Section 4.3. The *Expanded* methodology includes 93347 arrangements as possible solutions for the photovoltaic system. In contrast, in this approach, we reduce the number of arrangements to 24 only. In this approach, all verifiers above will solve the proposed photovoltaic systems and compute their respective results. Using this approach, we were able to avoid timeout and memory out problems. Thus, we can prove the technique's efficiency when the results are convergent to the same specification. In other words, each test case should have the same result when analyzed by the three SMT solvers.

Analyzing column number 2 of Table 1, we can observe that all benchmarks returned SAT as a response. This implies that there is a correct solution for each proposed case. Column 3 for the ESBMC and column 4 for the CPAchecker confirm that the results are convergent with the tool proposed here in search of a solution. However, the existence of a valid solution does not confirm that the proposed solution is the optimal solution. To confirm that PVz converges towards equipment specification, we must analyze the lines after SAT: NTP, NBT, controllers, inverters, and LCC. In the case of equality of the three tools, which we have successfully achieved, it can be inferred that the algorithm is sound. This means that the program works correctly and brings true results as an answer. In other words, that the **EG1** objective was successfully achieved. In the case of a difference between the results obtained, an incongruity in the solution developed.

Once it has been confirmed that the EG1 objective has been achieved, we must note that the tools' crucial difference is the processing time. PVz obtained interesting results, being approximately 13 times faster if compared to ESBMC and hundreds of thousands of times faster than CPAchecker. Our reduced approach solution proved to be robust against two of the state-of-the-art software verifiers.

Expanded approach: Once we have the solutions of the benchmarks found by the three aforementioned solvers, we focus on the objectives EG02 and EG03, respectively. In contrast to the first approach, the amount of equipment has increased considerably; this implies an increase in the computational cost to find the optimal solution.

For objective EG02, we analyzed the proposed technique's performance versus ESBMC, CPAchecker, and HOMER Pro tools. In Table 1, we can infer that both the ESBMC and CPAchecker tools failed to find a feasible solution for all the proposed simulations. HOMER Pro, obtained all the results except for the third case described in Table 1 column 08. For objective EG03, we make a comparison between columns 05 and 08. We conclude that the technique we have implemented is more accurate concerning the combinations of electrical equipment and their respective prices and LCC. Simultaneously, HOMER Pro, which we will describe in more detail in the next paragraph, processes the system a few seconds faster with an acceptable answer, but that is not the optimal one.

HOMER Pro: HOMER Pro was able to evaluate six case studies (cases 1, 2, 4, 5, 6, and 7) under 30 seconds. The test case 3 could not be simulated since HOMER Pro does not have the battery autonomy adjustment feature, i.e., the tool always tries to feed the given load with electricity 365 days/year. Some HOMER Pro drawbacks were also noted. (1) System equipment does not include an explicit charge controller. HOMER Pro includes a controller automatically to simulate the charge/discharge of batteries and meet the load requirement. However, without costs or electrical characteristics such as maximum current and voltage, which are common during PV sizing. (2) HOMER Pro requires the inclusion of some battery specifications to initiate optimization; however, it does not change the electrical specifications during simulation; the results presented are multiples of the original battery type suggested by the user. For example, it was started with an 83.4 Ah lead-acid battery, and during simulation, HOMER Pro did not try to use other capacities or types. (3) HOMER Pro does not present the optimal solution in terms of connections of PV panel arrays, just the total in terms of power, i.e., it presents neither the models and the power of each PV panel nor the total of panels in series or parallel. The cost of every equipment item used in HOMER Pro is a USA-based cost, without adaptation regardless of where the equipment is installed.

We have real PV systems deployed since June 2018 in a riverside community in the State of Amazonas, Brazil, GPS coordinates 2^{o}44'50.0"S 60^{o}25'47.8"W, with demands of case studies 1, 4, 5, and 6, always with a 3×325 W (3S, total 975 W) panels and 4×220 Ah (2S-2P $= 440$ Ah) lead-acid batteries.

TABLE 1
Comparative result table between state-of-the-art tools.

Tools	Microsoft Z3 (νZ 4.8.9)	ESBMC 6.2.0 (Boolector 3.0.1)	CPAchecker 2.0 (MathSAT 5.6.6)	Microsoft Z3 (νZ 4.8.9)	ESBMC 6.2.0 (Boolector 3.0.1)	CPAchecker 2.0 (MathSAT 5.6.6)	HOMER Pro 3.13.1
Test Case	Reduced Approach			Expanded Approach			Simulation
Case Study 1 Peak:342W Surge:342W E:3,900Wh/day Autonomy:48h	SAT (0,008 min) NTP:4×425w (2S-2P) NBT:8×220Ah Controller 30A/100V Inverter 400W/24V LCC: US$ 8,992.58	SAT (0,033 min) NTP:4×425w (2S-2P) NBT:8×220Ah Controller 30A/100V Inverter 400W/24V LCC: US$ 8,992.58	SAT (143,32 min) NTP:4×425w (2S-2P) NBT:8×220Ah Controller 30A/100V Inverter 400W/24V LCC: US$ 8,992.58	SAT (1,38 min) NTP:4×425w (2S-2P) NBT:8×220Ah Controller 30A/100V Inverter 400W/24V LCC: US$ 8,992.58	MO	MO	(Time: 0.33 min) 2.53 kW of PV NBT:12×83.4Ah (2S-6P) 0.351kW inverter LCC: US$ 7,808.04
Case Study 2 Peak:814W Surge:980W E:4,880Wh/day Autonomy:48h	SAT (0,003 min) NTP:6×355w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 1200W/24V LCC: US$ 10,028.34	SAT (0,05 min) NTP:6×355w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 1200W/24V LCC: US$ 10,028.34	SAT (143,35 min) NTP:6×355w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 1200W/24V LCC: US$ 10,028.34	SAT (1,48 min) NTP:6×355w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 1200W/24V LCC: US$ 10,028.34	MO	MO	(Time: 0.18 min) 3.71 kW of PV NBT:20×83.4Ah (2S-10P) 0.817kW inverter LCC: US$ 12,861.75
Case Study 3 Peak:815W Surge:980W E:4,880Wh/day Autonomy:12h	SAT (0,003 min) NTP:6×355w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 1200W/24V LCC: US$ 10,028.34	SAT (0,05 min) NTP:6×355w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 1200W/24V LCC: US$ 10,028.34	SAT (164,46 min) NTP:6×355w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 1200W/24V LCC: US$ 10,028.34	SAT (1,67 min) NTP:6×355w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 1200W/24V LCC: US$ 10,028.34	MO	MO	NA
Case Study 4 Peak:253W Surge:722W E:3,600Wh/day Autonomy:48h	SAT (0,003 min) NTP:4×395w (2S-2P) NBT:8×220Ah Controller 20A/100V Inverter 280W/24V LCC: US$ 8,883.62	SAT (0,05 min) NTP:4×395w (2S-2P) NBT:8×220Ah Controller 20A/100V Inverter 280W/24V LCC: US$ 8,883.62	SAT (144,7 min) NTP:4×395w (2S-2P) NBT:8×220Ah Controller 20A/100V Inverter 280W/24V LCC: US$ 8,883.62	SAT (1,42 min) NTP:4×395w (2S-2P) NBT:8×220Ah Controller 20A/100V Inverter 280W/24V LCC: US$ 8,883.62	MO	MO	(Time: 0.23 min) 2.42 kW of PV NBT:12×83.4Ah (2S-6P) 0.254kW inverter LCC: US$ 7,677.95
Case Study 5 Peak:263W Surge:732W E:2,500Wh/day Autonomy:48h	SAT (0,003 min) NTP:4×330w (2S-2P) NBT:6×220Ah Controller 20A/100V Inverter 280W/24V LCC: US$ 8,119.74	SAT (0,033 min) NTP:4×330w (2S-2P) NBT:6×220Ah Controller 20A/100V Inverter 280W/24V LCC: US$ 8,119.74	SAT (154,60 min) NTP:4×330w (2S-2P) NBT:6×220Ah Controller 20A/100V Inverter 280W/24V LCC: US$ 8,119.74	SAT (1,51 min) NTP:4×330w (2S-2P) NBT:6×220Ah Controller 20A/100V Inverter 280W/24V LCC: US$ 8,119.74	MO	MO	(Time: 0.18 min) 1.59 kW of PV NBT:10×83.4Ah (2S-5P) 0.268kW inverter LCC: US$ 6,175.57
Case Study 6 Peak:322W Surge:896W E:4,300Wh/day Autonomy:48h	SAT (0,003 min) NTP:6×330w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 400W/24V LCC: US$ 9,587.01	SAT (0,05 min) NTP:6×330w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 400W/24V LCC: US$ 9,587.01	SAT (153,42 min) NTP:6×330w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 400W/24V LCC: US$ 9,587.01	SAT (1,38 min) NTP:6×330w (3S-2P) NBT:10×220Ah Controller 40A/150V Inverter 400W/24V LCC: US$ 9,587.01	MO	MO	(Time: 0.22 min) 3.15 kW of PV NBT:14×83.4Ah (2S-7P) 0.328kW inverter LCC: US$ 9,112.45
Case Study 7 Peak:1,586W Surge:2,900W E:14,000Wh/day Autonomy:48h	SAT (0,003 min) NTP:20×330w (10S-2P) NBT:30×220Ah Controller 100A/400V Inverter 1600W/24V LCC: US$ 18,408.27	SAT (0,016 min) NTP:20×330w (10S-2P) NBT:30×220Ah Controller 100A/400V Inverter 1600W/24V LCC: US$ 18,408.27	SAT (130,08 min) NTP:20×330w (10S-2P) NBT:30×220Ah Controller 100A/400V Inverter 1600W/24V LCC: US$ 18,408.27	SAT (1,73 min) NTP:20×330w (10S-2P) NBT:30×220Ah Controller 100A/400V Inverter 1600W/24V LCC: US$ 18,408.27	MO	MO	(Time: 0.20 min) 12.5 kW of PV NBT:66×83.4Ah (2S-33P) 1.60kW inverter LCC: US$ 41,878.11

Legend: MO = memory out; TO = time out; E = energy; NTP = total number of panels, NBT = total number of batteries, S = in series; P = in parallel, LCC = Life Cycle Cost; NA = not available.

4.6 Comparison Between Formal Synthesis (νZ) and HOMER Pro

Let us consider νZ versus HOMER Pro since ESBMC and CPAchecker do not find an optimal solution in a typical scenario. In other words, this topic represents the second approach described previously. Let us compare the formal synthesis results against those of HOMER Pro. We consider that each database's cost of individual items used to compose the optimal design is not the same among the tools. As a result, it is plausible to obtain different results. Thus, we observed some distinct effects in terms of the technical solution and cost (cf. Table 1).

Regarding the processing time, both solved all the case studies in less than two minutes, which shows a significant advance concerning the other tools mentioned here. HOMER Pro solves the seven benchmarks three times faster than PVz, in counterpart to the performance, HOMER Pro does not return the global optimum cost, just an approximation, in our scenario, we find that this discrepancy is around 5% (financially, this variation is US $ 450 dollars) upwards or downwards. We took the seven benchmarks to obtain at these numbers, so we removed the third benchmark since HOMER Pro did not find a solution. The average price (LCC) was made by removing the benchmark with higher cost and lower cost.

Those discrepancies are not easy to address without some real systems validation. However, we use the simulation software PVsyst to validate the optimal sizing produced, as shown in Table 2. Note that PVsyst has a presizing feature, which presents a minimum recommended sizing of PV panels and batteries (only) without using manufacturers' data or models for it. This feature was used as a reference mainly with HOMER Pro, where there exist no equipment brands or models (only power and capacities specification). PVsyst was used with the field-deployed and the formal synthesis sizing solutions, where brands and models were simulated in PVsyst according to the sized system. Each simulation with PVsyst took 4 seconds. We were unable to validate the case study 3 using PVsyst. The battery autonomy is less than 24 hours, and only the proposed synthesis technique can perform the optimal sizing (PVsyst and HOMER Pro are limited for a 24 h minimum).

Overall, those comparisons with our approach, the optimization software, and the deployed systems, with validation through simulation tool, show that the synthesis solution is sound and complete, which answers *EG1* and EG3.

Concerning the cost (LCC) present by both tools, HOMER Pro does not use the real cost for PV systems deployed in Brazil; therefore, the optimal solution presented by HOMER Pro tends to be cheaper than our technique. However, considering that the aim is to present an optimal PV sizing solution that is feasible and closer to the market prices, our technique is more indicated.

Besides that, HOMER Pro suggests a value in kW for the inverters that are very close to every case study's maximum load, but it is not commercial. The proposed synthesis tool, however, presents inverters that are commercial and can be obtained off-the-shelf. Moreover, our synthesis approach considers surge power demand from the house, which HOMER Pro or PVsyst does not view. This feature is a definite advantage of the formal synthesis method. HOMER Pro does not include charge controllers as a specific equipment item in its mathematical model; only the synthesis tool presents a commercial controller and includes it during the cost analysis. The formal synthesis method, therefore, presents more reliable results than HOMER Pro.

Our synthesis technique can present a far more detailed solution and closer to commercial conditions than the answer given by HOMER Pro. In particular, the automated synthesis method can provide all the details of every component of a PV system solution, with complete electrical information from the manufacturer datasheet, including the component model, nominal current, and voltage. In this respect, even the manufacturer's name can be cited (in Table 1, we removed to avoid unauthorized advertising). Moreover, the validation through PVsyst simulation, using the PV sizing produced by HOMER Pro and our synthesis approach, shows that our results are feasible and not as oversized as HOMER Pro results, mainly concerning PV panels.

An optimal solution from a tool is not necessarily the same optimal from other tools, mainly when the database of equipment items (with different costs) is not the same [22]. Therefore, the comparison must take this issue into account.

4.7 Threats to validity

It is worth highlighting some points of attention. (i) in order to increase the technique's accuracy, it is necessary to increase the equipment database. However, this directly affects the complexity of computational processing. (ii) All costs (LCC) are based on local reality. It is necessary to adapt the inputs to support solar radiation, annual maintenance costs, and equipment costs from other countries and cities. (iii) The factors determining the benchmarks' energy demand are constant elements in the technique described here and ignore the different seasons, with specific temperature and solar irradiation changes. (IV) Lastly, the results described in the benchmarks must be deployed in the field, under real conditions of use, to obtain a final validation of the sized systems.

5 Conclusions

PVz proved to be an extremely efficient tool, optimizing equipment combinations for photovoltaic systems with higher performance for precision and verification time. Very close to the market tools like HOMER Pro. PVz withstood an exhaustive data load and concluded all benchmarks by delivering an optimal solution. The same has not happened with ESBMC and CPAchecker, which are state-of-the-art in software verification.

With the algorithms and methods developed in this work, we guarantee all benchmarks' correctness since all three verifiers could find the optimal solution. Besides, we built the tool to process equipment data on a much larger scale and always focus on the electrical system's best solution. Our advances are complemented by the extension of

TABLE 2
Optimal sizing validation with PVsyst.

CS	PVsyst (pre-sizing)	Field deployed validation	Formal synthesis sizing validation	HOMER Pro sizing validation
CS 1	P= 1,166 W B= 381 Ah (minimum)	Not correct sizing Avail. $<$ 95% (91.06%)	No error found 100% of avail.	No error found Panels oversized in 2.16 $\times$ Batteries oversized in 1.39 $\times$
CS 2	P= 1,482 W B= 478 Ah (minimum)	NA There exists no real PV system available for comparison	No error found 95.76% of avail.	No error found Panels oversized in 2.6 $\times$ Batteries oversized in 1.74 $\times$
CS 3	Not possible to simulate (autonomy $<$ 24h)	NA There exists no real PV system available for comparison	Only technique that produced solution	NA (autonomy $<$ 24h)
CS 4	P= 1,078 W B= 354 Ah (minimum)	No error found 95.76% of avail.	No error found 98.10% of avail.	No error found Panels oversized in 2.24 $\times$ Batteries oversized in 1.41 $\times$
CS 5	P= 823 W B= 268 Ah (minimum)	No error found 100% of avail.	No error found 100% of avail.	No error found Panels oversized in 1.93 $\times$ Batteries oversized in 1.56 $\times$
CS 6	P= 1,299 W B= 421 Ah (minimum)	Not correct sizing Avail. $<$ 95% (85.65%)	No error found 100% of avail.	No error found Panels oversized in 2.42 $\times$ Batteries oversized in 1.38 $\times$
CS 7	P= 4,263 W B= 1,384 Ah (minimum)	NA There exists no real PV system available for comparison	No error found 98.37% of avail.	No error found Panels oversized in 2.9 $\times$ Batteries oversized in 1.99 $\times$

Legend: CS = case study; NA = sizing not available for validation; B = batteries capacity; P = panels power; Avail.= Availability (expected of 95% or greater as a design requirement).

the CEGIS synthesis method implemented in the proposed tool. Lastly, we will apply the knowledge developed in this article to real riverside communities.

REFERENCES

[1] T. S. 7, "The energy progress report 2020," *2020 International Bank for Reconstruction and Development / The World Bank*, 2020.

[2] M. Hussein and W. LEAL FILHO, "Analysis of energy as a precondition for improvement of living conditions and poverty reduction in sub-Saharan Africa," in *Scientific Research and Essays*, vol. 7(30), 2012, pp. 2656–2666.

[3] S. Coelho, A. Sanches-Pereira, L. Tudeschini, J. Escobar, M. Poveda, N. Coluna, A. Collin, E. L. Rovere, A. Trindade, and O. Pereira, "Biomass residues as electricity generation source in low HD source in regions of Brazil," in *The XI Latin. Cong. of Elec. Gener. and Transm. CLAGTEE*, UNESP, Ed., 2015, pp. 1–8.

[4] S. Karekesi, K. Lata, and S. Coelho, *Renewable Energy - A Global Review of Technologies, Policies and Markets*. London: Earthscan, 2006, ch. Traditional Biomass Energy: Improving Its Use and Moving to Modern Energy Use, pp. 231–261.

[5] A. Benatiallah, D. Benatiallah, T. Ghaitaoui, A. Harrouz, and S. Mansouri, "Modelling and simulation of renewable energy systems in Algeria," *Int. J. of Sc. and App. Inf. Tech*, vol. 7, no. 1, pp. 17–22, 2017.

[6] S. Pradhan, S. Singh, M. Choudhury, and D. Dwivedy, "Study of cost analysis and emission analysis for grid connected PV systems using RETSCREEN 4 simulation software," *Int. J. of Eng. Res. & Tech.*, vol. 4, no. 4, pp. 203–207, 2015.

[7] N. Swarnkar, L. Gidwani, and R. Sharma, "An application of HOMER Pro in optimization of hybrid energy system for electrification of technical institute," in *Int. Conf. on Energ. Eff. Tech. for Sust. (ICEETS)*, 2016, pp. 56–61.

[8] A. Trindade and L. C. Cordeiro, "Optimal sizing of stand-alone solar PV systems via automated formal synthesis," *CoRR*, vol. abs/1909.13139, 2019. [Online]. Available: http://arxiv.org/abs/1909.13139

[9] V. Applasamy, "Cost evaluation of a stand-alone residential photovoltaic power system in malaysia," *2011 IEEE Symposium on Business, Engineering and Industrial Applications (ISBEIA)*, pp. 214–218, 2011.

[10] A. Abate, C. David, P. Kesseli, D. Kroening, and E. Polgreen, "Counterexample guided inductive synthesis modulo theories," in *Computer Aided Verification*, H. Chockler and G. Weissenbacher, Eds. Cham: Springer International Publishing, 2018, pp. 270–288.

[11] N. Bjørner, A. Phan, and L. Fleckenstein, "νz - an optimizing SMT solver," in *21st International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, ser. LNCS, C. Baier and C. Tinelli, Eds., vol. 9035. Springer, 2015, pp. 194–199.

[12] Y. Driouich, M. Parente, and E. Tronci, "A methodology for a complete simulation of cyber-physical energy systems," in *IEEE Work. on Envir., Energ., and Struc. Monit. Syst. (EESMS)*, 2018, pp. 1–5.

[13] M. Gadelha, F. Monteiro, J. Morse, L. Cordeiro, B. Fischer, and D. Nicole, "ESBMC 5.0: An industrial-strength C model checker," in 33^{rd} *ACM/IEEE Int. Conf. on Aut. Soft. Engin. (ASE'18)*. New York, NY, USA: ACM, 2018, pp. 888–891.

[14] D. Beyer and M. E. Keremoglu, "CPAchecker: A tool for configurable software verification," in *Lecture Notes in Computer Science*, G. Gopalakrishnan and S. Qadeer, Eds., vol. LNCS 6806. Springer, Berlin, Heidelberg, 2011, pp. 184–190.

[15] A. Trindade and L. C. Cordeiro, "Synthesis of solar photovoltaic systems: Optimal sizing comparison," in *Software Verification*, M. Christakis, N. Polikarpova, P. S. Duggirala, and P. Schrammel, Eds. Cham: Springer International Publishing, 2020, pp. 87–105.

[16] A. B. Trindade and L. C. Cordeiro, "Automated formal verification of stand-alone solar photovoltaic systems," *Solar Energy*, vol. 193, no. 1, pp. 684–691, 2019.

[17] R. F. Araujo, I. Bessa, L. C. Cordeiro, and J. E. C. Filho, "Smt-based verification applied to non-convex optimization problems," in *VI Brazilian Symposium on Computing Systems Engineering, SBESC 2026, João Pessoa, Paraíba, Brazil, November 1-4, 2016*. IEEE Computer Society, 2016, pp. 1–8.

[18] A. Abate, I. Bessa, D. Cattaruzza, L. Cordeiro, C. David, P. Kesseli, D. Kroening, and E. Polgreen, "Automated formal synthesis of digital controllers for state-space physical plants," in *Comp. Aided Verif. (CAV)*, vol. LNCS 10426, 2017, pp. 462–482.

[19] A. Hansen, P. Sørensen, L. Hansen, and H. Bindner, *Models for a stand-alone PV system*, ser. Denmark. Forskningscenter Risoe. Risoe-r. Forskningscenter Risoe, 2001, no. 1219.

[20] J. Pinho and M. Galdino, *Manual de Engenharia para Sistemas Fotovoltaicos*. Rio de Janeiro/RJ: CEPEL – CRESESB, 2014.

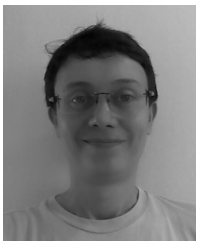
[21] M. Y. R. Gadelha, L. C. Cordeiro, and D. A. Nicole, "An efficient floating-point bit-blasting API for verifying C programs," *CoRR*, vol. abs/2004.12699, 2020. [Online]. Available: https://arxiv.org/abs/2004.12699

[22] T. Alsadi, S.; Khatib, "Photovoltaic power systems optimization research status: A review of criteria, constrains, models, techniques, and software tools," *Appl. Sci.*, vol. 8, no. 1761, pp. 1–30, 2018.

[23] A. Trindade, "Ferramenta de análise comparativa de projetos de eletrificação rural com fontes renováveis de energia na amazônia,"

in *IX Congresso sobre Geração Distribuída e Energia no Meio Rural - AGRENER GD*, 2013, p. n.pag.
[24] R. Brummayer and A. Biere, "Boolector: An efficient SMT solver for bit-vectors and arrays," in *Tools and Alg. for the Const. and An. of Sys. (TACAS)*, vol. LNCS 5505, 2009, pp. 174–177.
[25] A. Cimatti, A. Griggio, B. Schaafsma, and R. Sebastiani, "The MathSAT5 SMT Solver," in *Proceedings of TACAS*, ser. LNCS, N. Piterman and S. Smolka, Eds., vol. 7795. Springer, 2013, pp. 93–107.
[26] A. Trindade and L. C. Cordeiro, "Optimal sizing of stand-alone solar PV systems via automated formal synthesis," http://arxiv.org/abs/1909.13139, 2019.
[27] HOMER, "The HOMER microgrid software," http://www.homerenergy.com/software.html, 2017, [Accessed 1st June 2021].
[28] T. Khatib and W. Elmenreich, "Optimum availability of standalone photovoltaic power systems for remote housing electrification," *Int. Journal of Photoenergy*, vol. 2014, no. Article ID 475080, p. 5 pages, 2014.
[29] PVsyst, "Logiciel Photovoltaïque," https://www.pvsyst.com/, 2020, [Acessed 24 April 2020].
[30] S. Barua, R. A. Prasath, and D. Boruah, "Rooftop solar photovoltaic system design and assessment for the academic campus using PVsyst software," *International Journal of Electronics and Electrical Engineering*, vol. 5, no. 1, pp. 76–83, 2017.

Edilson Galvao received his computer engineering degree in 2015 from Foundation Center for Analysis, Research and Technological Innovation (FUCAPI) and is pursuing his M.Sc. at UFAM. Currently, he is a Software engineer at Samsung R&D. His interest is in automated verification and model checking. His work focuses on developing tools for process automation.



Alessandro Trindade received his Ph.D. in Computing, BSc and MSc in Electrical Engineering from the Federal University of Amazonas (UFAM) in 2020, 1995, and 2015, respectively. Currently, he holds an Adjunct Professor position in the Electricity Department from UFAM. Before joining UFAM, he worked four years as a renewable energy consultant to the Amazonas State Electric Utility and to the Inter-American Institute for Cooperation on Agriculture (IICA); he also worked for 12 years R&D and project manager at a non-profit foundation. His interest is in renewable energy, automated verification, and model checking.

Lucas Cordeiro received his Ph.D. degree in Computer Science in 2011 from the University of Southampton, UK. Currently, he is a Senior Lecturer in the Department of Computer Science at the University of Manchester, UK, and leads the Systems and Software Verification laboratory. He is also a collaborator in the Postgraduate Program in Electrical Engineering and Informatics at the Federal University of Amazonas (UFAM), Brazil. Before joining the University of Manchester, he worked as a researcher at Oxford University / Diffblue and as an assistant professor at UFAM; he also worked for six years as a software engineer in the industry. His work focuses on software model checking, automated testing, program synthesis, and embedded & cyber-physical systems.