Automated Verification of Stand-alone Solar Photovoltaic Systems

Alessandro Trindade[a,*], Lucas Cordeiro[b]

[a] *Federal University of Amazonas, Av. Rodrigo Octvio, 6200, Coroado I, 69077-000 Manaus-AM-Brazil*
[b] *University of Manchester, School of Computer Science, Kilburn Building, Manchester M13 9PL*

Abstract

With declining costs and increasing performance, the deployment of renewable energy systems is growing faster. In 2017, for the first time, the number of people without access to electricity dipped below 1 billion, but trends on energy access likewise fall short of global goals. Particular attention is given to stand-alone solar photovoltaic systems in rural areas or where grid extension is unfeasible. Tools to evaluate electrification projects are available, but they are based on simulations that do not cover all aspects of the design space. Automated verification using model checking has proven to be an effective technique to any kind of program verification. This paper marks the first application of software model checking to formally verify the design of a stand-alone solar photovoltaic system including solar panel, charge controller, battery, inverter, and electric load. Case studies, from real photovoltaic systems deployed in five different sites, ranging from 700W to 1,200W, were used to evaluate this proposed approach and to compare that with specialized simulation tools. Different verification tools were evaluated as well, in order to compare performance and soundness. Data from practical applications show the effectiveness of our approach, where specific conditions that lead to failures in a photovoltaic solar system are only detailed by the automated verification method.

*Corresponding author
Email address: alessandrotrindade@ufam.edu.br (Alessandro Trindade)

Keywords: Formal verification, model checking, photovoltaic power systems, power system modeling, solar power generation

1. Introduction

Lack of access to clean and affordable energy is considered a core dimension of poverty [1]. Progress has been made worldwide; in particular, the number of people without electricity access fell below 1 billion threshold for the first time in 2017 [2]. The share of people without access to electricity from Africa is 58%, while 19% of the share comes from developing Asia, and 31% from Latin America [2]. Numbers from Brazil show the aim to electrify 270 isolated places and 2.7 million of people until 2023 [3]. Furthermore, there is a close relationship between the lack of energy and the low HDI (Human Development Index) of those localities [4].

In order to provide universal 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 additional connections needed; and grid extension will be the standard especially in urban areas [2].

Only a niche market a few years ago, solar photovoltaic (PV) systems are now becoming a mainstream electricity provider, with an increase of approximately 50% from 2016 to 2017 in terms of new installations of PV [5]. However, this growth is not driven by isolated systems.

In order to simulate or evaluate a PV system there are a myriad of specialized tools available in the market, such as RETScreen, HOMER, PVWatts, SAM, and Hybrid2 [6, 7, 8, 9, 10]; and even general purpose simulation tools such as PSpice, Saber, or MATLAB/Simulink package [11, 12]. However, those tools are based on running experiments in simulation models. Simulation has the advantage of being cheap (if compared to test in real systems) and can be employed before the system design is concluded but it has the drawback of an incomplete coverage since the verification of all possible combinations and potential failures of a system is unfeasible [13].

Formal methods based on model checking offer a great potential to obtain a more effective and faster verification in the design process [13]. Any kind of system can be specified as computer programs using mathematical logic, which constitutes the intended (correct) behavior; then, one can try to give a formal proof or otherwise establish that the program meets its specification. In this study, a mathematical model of each component of a stand-alone PV system, as panel solar, charge controller, batteries, inverter, and electrical load is created. The behavior of each system component can be analyzed and observed with the support of those formal models, as a joint operation of the components, which in this case represents the operation of the solar PV system itself. The project requirements, as battery autonomy and power demand, besides weather conditions, as solar irradiance and ambient temperature, are translated as part of the computer program and automatically verified during the formal process. The model checking tool reports in which conditions a system does not meet the user requirements. A key benefit to this approach is that it helps in the detection of flaws in the design phase of system development, thereby considerably improving system reliability [14]. The implementation of the proposed tool is carried out by means of an algorithm written in language C, that is executed by three state-of-the-art model checkers to formally verifying PV designs, in order to evaluate performance and correctness: the C Bounded Model Checker (CBMC) [15], the Efficient SMT-based Bounded Model Checker (ESBMC) [16], and the Configurable Program Analysis Checker (CPAchecker).

In prior studies, the evaluation of PV systems w.r.t. user requirements were performed by simulation tools using MATLAB/Simulink [12, 17], or HOMER Pro [18]. Related to formal methods, studies were carried out toward the modeling of power smart grids [14] and solar panels [19]; however, those studies do not perform automated verification. In addition, recent research performed a formal study of PV farms, where the focus has been the interaction with the grid [20]; or to model a PV system and to verify the maximum power point of solar panels with the use of Jmodelica tool [21]. Both studies restrict to PV panels, and do not include batteries, inverters, and charge controllers.

This paper makes three original contributions. Firstly, we propose an algorithm written in language C that implements the automated verification method which formally checks the sizing and the operation of a given stand-alone PV system. Secondly, we evaluate the verification method by comparing three state-of-the-art model checkers in five real case studies. Thirdly, experimental results show that this proposed approach can find subtle design errors in stand-alone PV systems, which are not easily detected by other approaches based on simulation.

Outline. Section 2 gives the background about solar PV systems, design and validation of PV systems, and the mathematical modeling. Section 3 presents the automated verification technique. Section 4 shows how to perform the sizing of a stand-alone PV system. The methodology is presented in section 5. Section 6 is devoted to the results. Section 7 presents the conclusion and describes future work.

2. Solar Photovoltaic System

PV systems are classified into distinct types [22]. Specifically for remote rural areas of developing countries or places where the grid extension is not feasible, the most suitable configuration is the regulated stand-alone system with battery and AC load; this configuration is the focus of this study.

2.1. Design and Simulation of Solar PV systems

Here we evaluated the most popular tools available in the literature and summarized at Table 1: PVWatts, SAM, HOMER, RETScreen, and Hybrid2 [6, 7, 8, 9, 10].

As highlights, only Hybrid2 does not have technical support; HOMER and Hybrid2 perform off-grid system with battery backup analysis. Additionally, HOMER and RETScreen include economical analysis or even optimization-sensitive analysis. However, commercial version of those tools, called RETScreen Expert and HOMER Pro, are available only for Microsoft Windows and the annual subscription typically range from US$504.00 to US$657.00.

Table 1: Comparative coverage of reference software

Characteristic	PVWatts	SAM	HOMER	RETScreen	Hybrid2
Support	X	X	X	X	
Off-grid systems			X	X	X
Hybrid systems			X	X	X
Photovoltaics	X	X	X	X	X
Batteries			X		X
Main technical (T) or economical(E)	T	T	E	E	T
Optimization			X	X	
Sensitive analysis			X	X	

In this study, only HOMER has the minimum features for a comparative with our proposed verification approach. That evaluation is presented in Section 6.3.

2.2. Component models for a stand-alone PV system

A stand-alone PV system is illustrated in Fig.1. The PV generator (panel or array) is a semiconductor device that can convert solar energy into DC electricity. For night hours or rainy days, we need to hold batteries, where power can be stored and used. The use of battery as a storage form implies the presence of a charge controller [23]. 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 behavior of AC electrical load from the house that will be fed by the system.

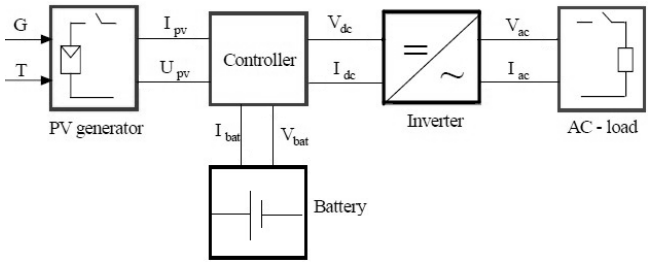

Figure 1: Block diagram for a typical stand-alone PV system [23].

2.3. PV generator model

A wide variety of models exists for estimation of power output of PV modules (and $I-V$ or $P-V$ curves). However, the present work will rely on the simplified model of 1-diode. It was demonstrated that it has a small error rate, between 0.03% and 4.68% from selected PV panels tested [24].

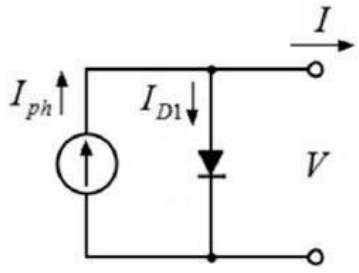

Figure 2: 1-diode equivalent PV cell/panel circuit model [25].

The 1-diode model, illustrated in Fig. 2, whose equation relates the output current, I, to the output voltage, V, is described by (1).

$$I = I_{ph} - I_{D1} = I_{ph} - I_0 \left[exp\left(\frac{V}{NaV_T}\right)\right], \tag{1}$$

where I_{ph} is the photocurrent delivered by the constant current source; I_0 is the reverse saturation current corresponding to the diode; N is the number of series-connected cells; a is the ideality or quality factor ($a = 1$ for ideal diodes and between 1 and 2 for real diodes); V_T is the thermal voltage ($V_T = k_B T/q$); k_B is the Boltzmann constant ($1.3806503 \times 10^{-23} J/K$); T the temperature of cell in Kelvin; q is absolute value of the electron's charge ($-1.60217646 \times 10^{-19} C$).

The voltage and the current at the maximum power point tracking (MPPT), can be described by (2), (3), and (4) as follows [24]:

$$V_m = \frac{aNk_BT}{q} ln\left(\frac{aNk_BT}{qI_0}\frac{I_{sc}}{V_{oc}}\right). \tag{2}$$

$$I_m = I_{ph} + I_0 - \frac{aNk_BT}{q}\left(\frac{I_{sc}}{V_{oc}}\right). \tag{3}$$

$$P_m = V_m I_m. \tag{4}$$

However, the photocurrent delivered by the constant current source (I_{ph}) or even the reverse saturation current (I_0) are not given by PV manufacturers. Therefore, (5) is used to calculate the photocurrent as function of irradiance and temperature [26]:

$$I_{ph} = \frac{G}{G_{ref}}\left[I_{ph,ref} + \mu_I\left(T - T_{ref}\right)\right], \tag{5}$$

where the reference state (STC) of the cell is given by the solar irradiance $G_{ref} = 1000W/m^2$ and the temperature $T_{ref} = 298.15K(= 25^oC)$; μ_I is the short-circuit current temperature coefficient (A/K) (provided by PV manufacturers). $I_{ph,ref}$ can be approximated to the reference short-circuit current [26] that is provided by PV manufacturers ($I_{sc,ref}$). The cell temperature (T) in degree Celsius is described by 6 [27]:

$$T = T_{air} + \frac{NOCT - 20}{800}G, \tag{6}$$

where T_{air} is the ambient temperature, $NOCT$ is the nominal operating cell temperature (in ^{o}C) that is found at the PV manufacturer's data-sheet [27], and G is the solar irradiance (W/m^2) of the place where the PV system is deployed.

Furthermore, (7) permits the saturation current (I_0) to be expressed as a function of the cell temperature as [26]

$$I_0 = \frac{I_{sc,ref} + \mu_I(T - T_{ref})}{exp\left[\frac{q(V_{oc,ref} + \mu_V(T - T_{ref}))}{aNk_BT}\right] - 1}, \tag{7}$$

where $V_{oc,ref}$ is the reference open-circuit voltage and μ_V is an open-circuit voltage temperature coefficient (V/K).

Using the maximum power point current (cf. (3)) and the saturation current in the reference temperature given by (7), the diode ideality factor [24] is determined by (8):

$$a = \frac{q(V_{m,ref} - V_{oc,ref})}{N k_B T} \frac{1}{ln\left(1 - \frac{I_{m,ref}}{I_{sc,ref}}\right)}, \tag{8}$$

where V_{mref}, $V_{oc,ref}$, $I_{m,ref}$, and $I_{sc,ref}$ are key cell values obtained under both actual cell temperature and solar irradiance conditions.

The PV generator model is now completely determined; it requires the actual cell temperature (or the air temperature), the actual solar irradiance and common data provided by manufacturers in data-sheets or manuals.

2.4. The Battery Storage Model

Various models have been described in the literature and the most common ones are based on lead-acid batteries [28, 29]; that kind of battery has relative low cost and wide availability [28]. Here, the model adopted uses only manufacturer's data without empirical tests [28]. The discharge voltage equation is described by (9).

$$V_d = [2.085 - 0.12(1 - SOC)] - \frac{I}{C_{10}} \left(\frac{4}{1 + I^{1.3}} + \frac{0.27}{SOC^{1.5}} + 0.02\right) (1 - 0.007\Delta T), \tag{9}$$

where C_{10} means 10h of rated capacity, which is standard on the manufacturer's data-sheet, ΔT is temperature variation ($\Delta T = T - T_{ref}$), SOC or state of charge indicates how much electric charge is stored in the cell at a given time. Mathematically, it is the ratio between the present capacity and the nominal capacity. If $SOC = 1$, then the battery is totally charged; and if $SOC = 0$, then the battery is fully discharged. The depth of discharge (DOD) or the fraction of discharge, is $DOC = 1 - SOC$.

For the charging process, however, the parameters are described by (10) as

$$V_c = [2{+}0.16SOC]{+}\frac{I}{C_{10}}\left(\frac{6}{1+I^{0.86}}+\frac{0.48}{(1-SOC)^{1.2}}+0.036\right)(1{-}0.025\Delta T). \tag{10}$$

Note that SOC can be calculated easily at any point during the discharge period, thereby considering the current drained from batteries during a certain time period.

2.5. Charge Controller Model

The charge controller or controller is a set of items (DC-DC converter, MPPT, and switches) and can be defined as the responsible to manage the energy flow to PV system, batteries and loads by collecting information on the battery voltage and knowing the maximum and minimum values acceptable for the battery voltage. MPPT, or maximum power point tracking, is an electronic control mechanism that maintains the PV operating at the voltage of maximum power. Controllers aim to protect the battery (or batteries) against the excessive charge and discharge [29], improving its lifetime.

To protect the battery against an excessive charge, the PV arrays are disconnected from the system, when the terminal voltage increases above a certain threshold V_{max_off} and when the current required by the load is less than the current delivered by the PV arrays [23]. PV arrays are connected again when the terminal voltage decreases below a certain value V_{max_on}. In order to protect the battery against excessive discharge, the load is disconnected when the terminal voltage falls below a certain threshold V_{min_off} and when the current required by the load is larger than the current delivered by the PV arrays [23]. The load is reconnected to the system, when the terminal voltage is above a certain value V_{min_on}. The steps in the modeling of the controller process are summarized in Table 2.

The output power (P_{out}) of controller is given by (11).

$$P_{in}\eta_c = P_{out}. \tag{11}$$

Table 2: Summary of the controller process (Source: [23])

Step	Constraint	Command
(1)	If $V > V_{max\_off}$ and $I_{load} < I_{pv}$	Disconnect PV array from the system
(2)	If command (1) is done and $V < V_{max\_on}$	Reconnect PV array to the system
(3)	If $V < V_{min\_off}$ and $I_{load} > I_{pv}$	Disconnect the load from the system
(4)	If command (3) is done and $V > V_{min\_on}$	Reconnect the load to the system

Assuming that the efficiency of the controller (η_c) is a manufacturer's data, from (11) we compute (12).

$$V_{in} I_{in} \eta_c = V_{out} I_{out}, \tag{12}$$

where V_{in} is the voltage across the PV array, I_{in} is the output current of PV array, V_{out} is the DC bus voltage, and I_{out} is the output current from the converter.

2.6. The inverter model

The role of the inverter is to keep the voltage constant on the AC side, i.e., at the rated voltage, and to convert the input power P_{in} into the output power P_{out} with the best possible efficiency η_i as described by (13) [23]:

$$\eta_i = \frac{P_{out}}{P_{in}} = \frac{V_{AC} I_{AC} cos\varphi}{V_{DC} I_{DC}}, \tag{13}$$

where I_{DC} is the current required by the inverter from the DC source to be able to keep the rated voltage on the AC side, V_{DC} is the input voltage to the inverter delivered by the DC source (PV panel or battery), V_{AC} and I_{AC} are the output voltage and current, respectively, and $cos\varphi$ can be obtained from the inverter's data-sheet.

2.7. Availability of Stand-alone PV Systems

Each stand-alone PV system, like any other power systems, has a specific level of availability. This reliability level impacts various issues, as system performance, production, feasibility, and investment. The availability of a stand-alone PV system can be defined as the percentage of time at which a power system is capable of meeting the load requirements [30]. The number of hours that the system is available, divided by 8,760 h, gives the annual system availability.

The system availability definition depends on how critical the load application is. For critical loads, 99% is considered acceptable. While in a ordinary house electrical load, 95% is considered acceptable. As an example, a system with 95% availability is expected to meet the load requirement of 8,322 h during an average year for the entire useful life of the system. An annual availability of 99% means that the system can operate the load for 8,672 h of the 8,760 h.

With that in mind, it is important to mention that even if a formal verification or a simulation shows that a PV system fails, it doesn't mean that the sizing is wrong. It is important to evaluate how critical the load application is and the horizon of evaluation. However this analysis can be useful, for example, to improve the sized battery autonomy.

3. Automated Verification Using Model Checking

Although simulation and testing explore possible behaviors and scenarios of a given system, they leave open the question of whether unexplored trajectories may contain a flaw. Formal verification conducts an exhaustive exploration of all possible behaviors; when a design is said to be "correct" by a formal verification method, it implies that all behaviors have been explored, and questions regarding adequate coverage or missed behavior becomes irrelevant [31]. Formal verification is a systematic approach that applies mathematical reasoning to obtain guarantees about the correctness of a system; one successful method in this domain is model checking [31]. Here we evaluate three state-of-the-art model checkers to formally verifying PV designs w.r.t. user requirements.

3.1. CBMC

The C Bounded Model Checker (CBMC) falsifies assertions in C programs or proves that they are safe if a completeness threshold is given [15]. CBMC implements a bit-precise translation of a C program, annotated with assertions and with loops unrolled up to a given depth, into a logical formula. If the formula is satisfiable, then a failing execution that leads to a violated assertion exists [15]. CBMC's verification flow can be summarized in three stages: (i) Front-end: scans, parses and type-checks C code; it converts control flow elements, such as *if* or *switch* statements, loops and jumps, into equivalent guarded *goto* statements, thus aiming to reduce verification effort; (ii) Middle-end: performs symbolic execution by eagerly unwinding loops up to a fixed bound, which can be specified by the user on a per-loop basis or globally, for all loops and finally; (iv) Back-end: supports SAT and SMT solvers to discharge verification conditions.

3.2. ESBMC

ESBMC (or Efficient SMT-based Bounded Model Checker) is a bounded and unbounded mdoel checker for C programs [16], which supports the verification of LTL properties with bounded traces [32]. ESBMC's verification flow can be summarized in three stages: (i) a front-end that can read and compile C code, where the system formal specification is first handled; (ii) preprocessing steps to deal with code representation, control flow and unwinding of loops, and model simplification, thereby aiming to reduce verification effort; and finally (iii) the SMT solving stage, where all constraints and properties of the system are encoded into SMT and checked for satisfiability. ESBMC exploits the standardized input language of SMT solvers (SMT-LIB[1] logic format) to make use of a resource called *assertion stack* [33]. This enables ESBMC, and the respective solver, to learn from previous checks, thus optimizing the search procedure and potentially eliminating a large amount of formula state space to

[1] http://smtlib.cs.uiowa.edu/

be searched, because it solves and disregards data during the process, incrementally. This technique is called "incremental SMT" [34] and allows ESBMC to reduce the memory overhead, mainly when the verified system is complex and the computing platform does not have large amount of memory to deal with the entire design space state.

3.3. CPAchecker

Configurable program analysis (CPA) provides a conceptual basis for expressing different verification approaches in the same formal setting. The CPA formalism provides an interface for the definition of program analyses; consequently, the tool CPAchecker provides an implementation framework that allows the seamless integration of program analyses that are expressed in the CPA framework. The comparison of different approaches in the same experimental setting is intended to be easy and the experimental results is expected to be more meaningful [35]. Related to the architecture, the central data structure is a set of control-flow automata (CFA), which consist of control-flow locations and control-flow edges. The CPA framework provides interfaces to SMT solvers and interpolation procedures, such that the CPA operators can be written in a concise and convenient way [35]. Currently CPachecker uses MathSAT as an SMT solver, and CSIsat and MathSAT as interpolation procedures [35]. The CPA algorithm performs the reachability analysis, and operates on an object of the abstract data type CPA, i.e., the algorithm applies operations from the CPA interface without knowing which concrete CPA it is analyzing. For most configurations, the concrete CPA will be a composite CPA, which implements the combination of different CPAs. In software verification, its usual to take a considerable amount of effort to convert a verification idea into actual experimental results and CPAchecker aim to accelerate this process [35].

4. Stand-alone PV System Sizing and Verification

In addition to the verification tool being able to check a PV solar system, as if it were actually being used in the field, indicating the conditions that it does

not meet the system user requirements, it is possible to extend the tool, giving it the capacity to carry out the validation of the designed system (a.k.a. sizing check).

Therefore this is a additional feature included at this paper. The sizing check will be performed by the proposed tool, and it will indicate if there is an error of sizing before to perform the automated verification of the system. This stage ensures that the system meets the standard project steps related to worst monthly average solar energy [29]. Firstly, it is needed to correct the energy consumption estimated to the load ($E_{consumption}$), which is carried out by (14), where the efficiency of batteries (η_b), controller (η_c), and the inverter (η_i) are considered [29].

$$E_{corrected} = \frac{E_{consumption}}{\eta_b \eta_c \eta_i}. \tag{14}$$

Following, it is necessary to estimate the energy that can be produced for each panel, called E_p, in Wh, as defined by (15).

$$E_p = Solar\_Irradiance \times Panel\_Area \times \eta_p \times 1000, \tag{15}$$

where the solar irradiance is expressed in terms of kWh/m^2 and depends on the site where the PV systems will be deployed, the PV panel area is given in m^2 and corresponds to the size of one PV panel, and η_p represents the PV panel efficiency.

The total minimum number of needed solar panels (N_{TPmin}) is computed by (16) and the check is performed using (17), where the sized number of panels (N_{TP}) must be greater than the result from (16).

$$N_{TPmin} = \frac{E_{corrected}}{E_p}. \tag{16}$$

$$N_{TP} \geq N_{TPmin}. \tag{17}$$

Particularly, the total number of panels in series (N_{PSmin}) and parallel (N_{PPmin}) are given by (18) and (19), respectively. With the check performed by (20) and (21). Where $V_{mppt,max}$ is the maximum operation voltage and

$V_{mppt,min}$ is the minimum operation voltage of the charge controller; $V_{maxPower,TempMax}$ and $V_{maxPower,TempMin}$ are the maximum power voltage from the PV module considering the maximum and minimum operational temperature, respectively; P_{total} is the total power demanded from the PV system and $P_{max,ref}$ is the power supplied from one PV panel in $Watts$.

$$\frac{V_{mppt,min}}{V_{maxPower,TempMax}} \leq N_{PSmin} \leq \frac{V_{mppt,max}}{V_{maxPower,TempMin}}. \tag{18}$$

$$N_{PPmin} = \frac{P_{total}}{Number\,Panels\,Series \times P_{max,ref}}. \tag{19}$$

$$N_{PS} \geq N_{PSmin}. \tag{20}$$

$$N_{PP} \geq N_{PPmin}. \tag{21}$$

Related to the batteries, firstly must be defined the total capacity of the battery bank, as described by (22) as

$$C_{bank} = \frac{E_{corrected} \times autonomy}{V_{system} \times DOD}. \tag{22}$$

where the variable *autonomy* is a design definition; V_{system} is the DC voltage of the bus, and the other variables were discussed previously in Section 2.3 and 2.4. Secondly, the total (minimum) number of batteries is computed, as described by (23). Additionally, (24) performs the final sizing check, thus considering the number of batteries in series (N_{BS}) and the number of batteries in parallel (N_{BP}) that are established to the project.

$$N_Btotal = N_{BS}min \times N_{BP}min = \frac{V_{system}}{V_{bat}} \times \frac{C_{bank}}{1\,Battery\,Capacity}. \tag{23}$$

$$(N_{BS} \times N_{BP}) \geq N_Btotal. \tag{24}$$

Related to the charge controller, initially it must meet the voltage requirement of the PV system, as described by (25) to the charge controller voltage:

$$V_c = V_{system}. \tag{25}$$

Following, the short circuit reference information from the manufacturer's solar panel must be corrected to the cell temperature, as described by (26):

$$I_{sc,amb} = \frac{G}{G_{ref}} \left[I_{sc,ref} + \mu_I \times (T - 25) \right]. \tag{26}$$

The controller must meet the maximum current from the PV array given by (27) and (28).

$$I_{c,min} = I_{sc,amb} \times N_{PP}. \tag{27}$$

$$I_c \geq I_{c,min}. \tag{28}$$

The sizing project check of the inverter is carried out by means of three equations. Equation (29) ensures that the input voltage of the controller meets the system voltage. Equation (30) ensures that the output voltage of the controller meets the AC voltage of the load. Finally, (31) ensures that the controller can support the total demand of the load and the surge power. Where $V_{in}DC$ is the nominal input voltage and $V_{out}AC$ is the nominal output voltage of the inverter; $MAX_{AC,ref}$ is the peak power that the inverter can support.

$$V_{in}DC = V_{system}. \tag{29}$$

$$V_{out}AC = V_{AC}. \tag{30}$$

$$\left[(Demand \leq P_{AC,ref}) \, and \, (P_{surge} \leq MAX_{AC,ref}) \right]. \tag{31}$$

Worth to mention that the charge controller, the battery, and the inverter have a nominal efficiency presented at data sheets, and those numbers are very close to 100%. However, based on empirical data, those efficiencies must be corrected, according to the loading of each component. If field or essay data is not available, considering that the proposed tools is aimed to be used prior

the deployment of the PV system at the field, it is recommended (at least) to correct the battery efficiency (η_b) to 85%, keeping the remaining efficiencies as expressed by the manufacturers.

5. Proposed Automated Verification Method of PV Systems

The flowchart of the proposed automated verification method is illustrated in Fig. 3.

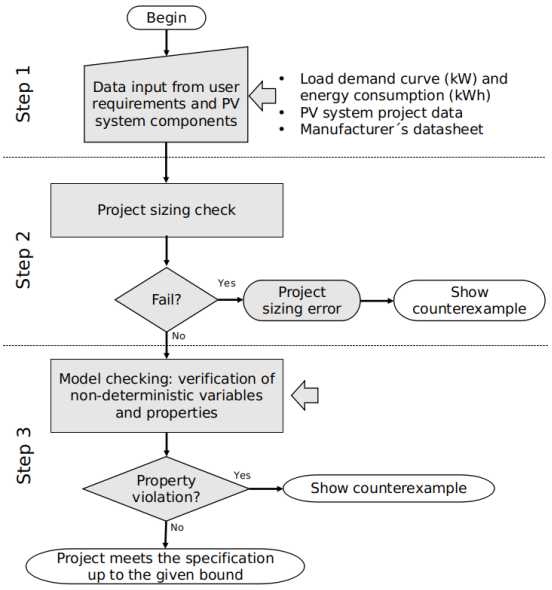

Figure 3: Flowchart of the proposed automated verification of PV systems.

In Step 1, the PV input data (e.g., load power demand and load energy consumption) and the formulae to check the sizing project, the mathematical model, the limits of the weather non-deterministic variables, are all written as an ANSI-C code [36]. In Step 2, the sizing check of the PV system takes place to make sure that the components were selected according to worst monthly average solar energy method [29]. The equations to check the PV sizing are present at Section 4. In Step 3, weather variables (e.g., solar irradiance and ambient temperature) will be systematically explored by our verification engine

based on maximum and minimum values from the site, where the PV system will be deployed. In addition, depending on one of the desired properties of the system such as battery autonomy, energy availability, or even system power supply, our verification engine is able to indicate a failure if those properties are not met; in this particular case, it provides a diagnostic counterexample that shows in which conditions the property violation occurred.

In a nutshell, the model checker will process the ANSI-C code with constraints and properties from the PV system that are provided by the user, and the tool will automatically verify if the PV system requirements are met. If it returns a failure (i.e., SAT), then the tool provides a counterexample, i.e., a sequence of states that leads to the property violation; this information can be used as a feedback to improve the PV system design. However, if the verification succeeds (i.e., UNSAT), there is no failure up to the bound k; therefore, the PV system will present its intended behavior up to the bound k.

Algorithm 1 describes the pseudo-code used to perform the automated verification. In order to reduce the computational effort of the algorithm, every 24h-day was considered as a time-step of 1 hour, and it was split into two parts: (a) one where it is possible to occur PV generation, during daylight, with a duration in hours depending on each site (but dependent on the sun and weather conditions); and (b) one that includes all the remaining day (without any PV generation), when the batteries are demanded to feed the house.

Lines 1 is devoted to information from the location where the PV system will be/were deployed. Usually the information of annual average minimum and maximum, related to temperature (T) and solar irradiance (G), hour by hour, are obtained from specific historical weather web sites [37], [38].

Line 2 represents all the information that comes from the PV sizing and from the equipment manufacturers data, as PV panels specification and data, batteries specification and data, inverter details and charge controller details. This item includes as well information from the house's load curve.

The first automated verification is related to the sizing check, according described at Section 4. Then two functions, called at lines 4 and 5, are respon-

sible for discover which hour starts the photovoltaic generation and when stops. Those functions get this information from the array inputted to the Algorithm with the solar irradiance values.

The batteries are assumed to have SOC of 100% (Line 6), i.e., the batteries are charged. That is important because the system is said ready to use when the batteries are charged.

The first for-loop at Line 7 controls how many cycles of 24h will be performed by the Algorithm. And the for-loop from lines 8 to 11 is responsible to discharge the battery (according the load curve) and verify the state of charge of the battery, hour-by-hour, starting at the first hour of the day after the sun goes down until the next day before the sun goes up (when there is not PV generation). Following, at the next for-loop, from line 12 to 29, is performed the verification where there is solar irradiance and all the PV system works. The Algorithm generates information related to average temperature (T) and solar irradiance (G), maximum and minimum annual, hour-by-hour, using non-deterministic variables from model checker to explore all possible states and the *assume* macro to constrain the non-deterministic values using a given range (lines 15 and 16).

After that, the model from PV generator is used in the function call of Line 17, to produce the voltage and current considering the states of G and T. With respect to every hour considered, the conditional *if-elseif-endif* statements from Lines 18, 20, 22, 24 and 26, will imitate the charge controller work, performing the charge or discharge of batteries according to the value of different variables: if there is PV generation, the updated state of charge from batteries, the house's load and the set-up information of the PV system.

At the end of last for-loop, the state of the batteries is verified again (Line 27) and the hour is adjusted to the next loop (Line 28).

Nevertheless, if the verification engine does not fail, then we can conclude that the PV system does not need further corrections up to the given bounds k.

Algorithm 1 Model checking algorithm for stand-alone PV

1: $declare\ min\ and\ max\ solar\ irradiation[24h],\ and\ temperature[24h]$
2: $declare\ case\ studies\ details:\ sizing\ and\ manufacturers\ data$
3: $sizing\_check()$
4: $startPVgeneration \leftarrow findStartPVgeneration()$
5: $endPVgeneration \leftarrow findEndPVgeneration()$
6: $SOC \leftarrow 100\%$
7: **for** $1st\ 24h\ loop$ to $Nth\ 24h\ loop$ **do**
8: **for** $endPVgeneration + 1$ to $startPVgeneration - 1$ **do**
9: $dischargeBattery\ in\ 1h()$
10: $assert(SOC \geq SOC\_min)$
11: **end for**
12: **for** $startPVgeneration$ to $endPVgeneration$ **do**
13: $G \leftarrow nondet\_uint(\ )$ {G is non-deterministic variable}
14: $T \leftarrow nondet\_uint(\ )$ {T is non-deterministic variable}
15: assume $(Gmin \leq G \leq Gmax)$ {restricting G values}
16: assume $(Tmin \leq T \leq Tmax)$ {restricting T values}
17: $Imax, Vmax \leftarrow PVgenerationMODEL(G, T)$
{If-then-else sequence to imitate charge controller work}
18: **if** $(battery\ is\ empty)$ AND $(PV\ is\ generating)$ **then**
19: $chargeBattery\ in\ 1h()$ {PV feed the house}
20: **else if** $(battery\ is\ empty)$ AND NOT$(PV\ is\ generating)$ **then**
21: FAIL with assert macro {Battery is empty and there is not PV generation}
22: **else if** NOT$(battery\ is\ empty)$ AND $(PV\ is\ generating)$ **then**
23: stop battery charge {PV feed the house}
24: **else if** NOT$(battery\ is\ empty)$ AND NOT$(PV\ is\ generating)$ **then**
25: $dischargeBattery\ in\ 1h()$ {Battery feed the house}
26: **end if**
27: $assert(SOC \geq SOC\_min)$
28: $hour \leftarrow hour + 1$
29: **end for**
30: **end for**
31: **return** ( )

Table 3: Case studies: stand-alone solar PV systems.

Item	House 1	House 2	House 3	House 4	House 5
PV Panels	3x 325W: in series				4x 325W: 2x series, 2x parallel
Batteries	4x220Ah: 2x series, 2x parallel autonomy: 48h				4x 120Ah batteries in series autonomy: 6h
Charge Controller	With MPPT of 150V/35A				
Inverter	700W with peak of 1,600W				1,200W with peak of 1,600W
Power demand	253W	263W	283W	501W	915W
GPS Coordinates	2^o44'50.0"S 60^o25'47.8"W				3^o4'20.208"S 60^o0'30.168"W
Details	Riverside indigenous community Rural Area of Manaus - Brazil				Urban house Manaus-Amazonas-Brazil

6. Experimental Evaluation

6.1. Description of the Case Studies

We have performed five case studies to evaluate the tools, as described in Table 3.

6.2. Objectives and Setup

Our experimental evaluation aims to answer two research questions:

RQ1 **(soundness)** Does the automated verification provide correct results?

RQ2 **(performance)** How do the verifiers compare to each other and to a simulation commercial tool?

In order to evaluate the proposed verification method and its performance, we have considered five case studies, three verification engines in different configurations, and also compared the results to the HOMER Pro tool. All the experiments were performed with a timeout of 14,400 seconds (4 hours).

All verification engines experiments were conducted on an otherwise idle Intel Xeon CPU E5-4617 (8-cores) with 2.90 GHz and 64 GB of RAM, running Ubuntu 16.04 LTS 64-bits. The setup of HOMER Pro simulation tool: Intel Core i5-4210 (4-cores), with 1.7 GHz and 4 GB of RAM, running Microsoft Windows 10; we have used HOMER Pro v3.12.0.

Verification engine ESBMC, version v6.0.0 was used with the SMT solver Boolector version 3.0.1 [39][2]; and an alternative ESBMC v6.0.0 was used with the SMT incremental mode[3] enabled with the goal of reducing memory usage; we have also used the SMT solver Z3 version 4.7.1 [40].

Verification engine CBMC 5.11 and MiniSat 2.2.1 were used in the comparation [15][4].

Verification engine CPAchecker 1.8 was used [5], with the SMT solver MathSAT version 5.5.3 [41]. An alternative CPAchecker configuration was tried as well, using BMC k-induction option, but without improvements of performance or soundness.

6.3. Results and Discussion

Table 4 summarizes the results concerning the use of the automated verification tools applied to the five case-studies of a solar PV system. The times reported in Table 4 answer RQ2.

Note that if we used HOMER Pro in all case studies, the $1,200$W was the only one that was proved to not meet the requirement of battery autonomy; all the 700W systems had no indication of flaws during simulation. The simulation took less than five seconds to be performed on each case study.

Worth to mention that a UNKNOWN result from the verification engines means that the result is not SAT or UNSAT. This result indicates that the verification created a out of memory or a time out situation. It is possible yet, that the UNKNOWN can be caused by user interruption, however this is not an option during the tests performed during the experimentation.

[2]Command-line: $ esbmc filename.c --no-bounds-check --no-pointer-check --unwind 100 --boolector

[3]Command-line: $ esbmc filename.c --no-bounds-check --no-pointer-check --unwind 100 --smt-during-symex --smt-symex-guard --z3

[4]Command-line: $ cbmc filename.c –unwind 100 –trace

[5]Command-line: $ scripts/cpa.sh -heap 64000m -stack 10240k -config config/bmc-incremental.properties -spec config/specification/sv-comp-reachability.spc filename.c

Table 4: Summary of the case-studies comparative and the automated tools.

Model Checker (SAT/UNSAT: time and message)				
Case	ESBMC 6.0.0 (Boolector 3.0.1)	ESBMC 6.0.0 (Z3 4.7.1)	CBMC 5.11 (MiniSat 2.2.1)	CPAchecker 1.8 (MathSAT 5.5.3)
House 1	Out of memory (UNKNOWN)	5m08s (UNSAT)	19m02s (UNSAT)	Time out (UNKNOWN)
House 2	Out of memory (UNKNOWN)	4m27s (UNSAT)	18m59s (UNSAT)	Time out (UNKNOWN)
House 3	Out of memory (UNKNOWN)	5m07s (UNSAT)	18m39s (UNSAT)	Time out (UNKNOWN)
House 4	Out of memory (UNKNOWN)	4m37s (UNSAT)	18m36s (UNSAT)	Time out (UNKNOWN)
House 5	Out of memory (UNKNOWN)	$\leq$ 1 sec (SAT Line 337)	$\leq$ 1 sec (SAT Line 337)	6 sec (SAT line 337)

The description of our experimental results can be broken down into three parts, one to each verification engine: ESBMC, CBMC, and CPAchecker.

Related to the ESBMC, it was tried two possibilities, with the solvers Boolector and with Z3. The incremental option, that uses less memory, can only be performed with the solver Z3. Using ESBMC with the solver Boolector created a out of memory situation in all the case studies. And this result was obtained in less than six minutes of execution, i.e., the 64 G bytes of RAM were consumed by the verification engine and the processes was killed. This resulted in a UNKNOWN return from the tool, as shown at the first column in 4. However, using the same version of the ESBMC but with the Z3 solver, the experimentation returned SAT or UNSAT to all the case studies. Related to the cases that use a 700 W PV system, the tool could not reach an error in all the four houses and the execution time took from 4m27s to 5m08s. However the 1,200 W PV system (house 5) failed (SAT) at the line 337 of the code, indicating that the system is *incorrectly* sized; in particular, the counterexample provided by the verification engine indicated that the nominal current from the charge controller is less than the minimum current demanded by the PV system in order to work properly. This verification took less than 1s to be performed.

Concerning the CBMC tool, similar results were obtained, but with some

slower time. The experimentation returned SAT or UNSAT to all the case studies. Related to the 700 W PV systems, the tool could not reach an error in all the four houses and the execution time took from 18m36s to 19m02s. However the 1,200 W PV system (house 5) failed (SAT) at the line 337 of the code; with the same counterexample presented by ESBMC. This verification took less than 1s to be performed as well.

Finally, the CPAchecker tool presented some different results. Even using two different configuration possibilities, as described in Section 6.2, the verification engine presented a UNKNOWN result for all the 700W systems. This is because the time out set up was reached, i.e., after 4 hours of execution the tool was not possible to decide if the verification was SAT or UNSAT. However, when verifying the 1,200W PV system, the tool presented the same SAT message, indicating the same line (337) of the code.

In order to validate the possible flaw from the house 5, presented by all verification engines, it was surveyed the owner of the 1,200 W system; we identified that, in fact, the system does not meet the battery autonomy most of the time (mainly when all loads are turned on), and this was double checked with the monitoring system from the charge controller from the manufacturer who showed that the maximum power or surge power were not exceeded, thus affirming RQ1; this behavior is expected since the system was purchased as an off-the-shelf solution and not as a customized design for the electrical charges of the house. The same process of validation was done to the four 700 W PV systems: from July of 2018 to March 2019, a monthly visiting was performed to apply a survey to the dwellers and to collect data from a local monitoring system with the Raspberry Pi controller: not every month the dwellers reported some energy interruption of the PV system, normally in raining or clouds days, thereby reaching the situation described in Step 3 of Table 2. And even when a interruption is reported, this represents around 3.33% of interruption in one month, what represents 96.97% of availability of the PV system and it is in accordance to what was described in Section 2.7; and that is not considered a flaw, further affirming RQ1.

The same five case studies were evaluated by HOMER Pro (RQ2). The simulation results showed that the project restrictions were met by four 700 W PV systems (house 1, 2, 3 and 4), without any indication of sizing error or even performance related issues. The case study that was unsuccessful during simulation was the 1,200 W (house 5); however, without any indication about the failures of this PV system (RQ2). All the simulations took less than 5 seconds (each) to be performed by HOMER Pro. Fig. 4 shows one of the screens presented by HOMER Pro software, specifically to house 1.

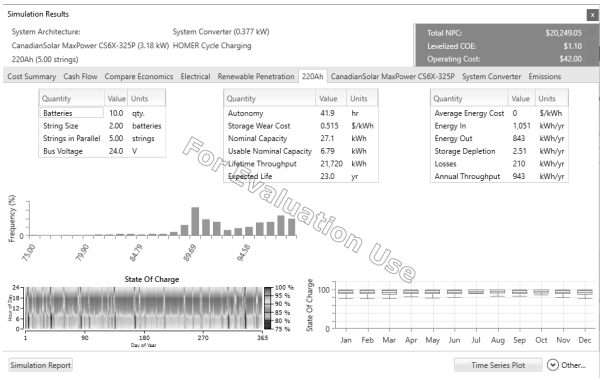

Figure 4: Homer simulation screen.

There were no divergence of results for the houses 1, 2, 3 and 4 w.r.t. our proposed approach, it is evident that the information collected from the dwellers and from the monitoring systems indicate that our approach provides the correct evaluation of the PV system, thus answering RQ2. House 5 presented flaws from all tools (automated verified or simulation); however, only automated verification approaches indicated which design error was responsible for the flaw (charge controller specification), further answering RQ2.

6.4. Threats to Validity

We have reported a favorable assessment of the proposed method. Nevertheless, we have also identified three threats to the validity of our results that can further be assessed.

Model precision: each component of the PV system is mathematically modeled. A careful evaluation in a PV laboratory to validate the model could add more reliability to the results produced by our method.

Time step: The run-time complexity of our proposed method is an issue; the time step of one hour can be further reduced to approximate the algorithm to the real-world scenario, where a solar irradiance and ambient temperature can change in fractions of minutes.

Case studies: Our case studies are performed only in one municipality. A more complete evaluation can be performed if other places around the world could become case studies.

7. Conclusions and Future Work

We have described and evaluated an automated verification method to check whether a given PV system meets its specification using software model checking techniques. We have considered five case studies from real photovoltaic systems deployed in five different sites, ranging from 700 W to $1,200$ W; and three state-of-art verification engines were considered (ESBMC, CBMC, and CPAchecker). Although the verification method proposed takes longer than simulation methods, it is able to present details (counterexample) that lead to failures in a PV system that is not a feature presented in commercial simulation tool. In particular, the proposed method was successful in finding sizing errors and indicating in which conditions a PV system can fail. Related to the verification engines comparative, the ESBMC tool with the Z3 solver executed in the incremental configuration presented the better performance (around four times faster than CBMC), used less RAM memory (less than 2G bytes when compared to 9.2G bytes of CBMC and 19.2G bytes of CPAchecker), and all the results were correct because the simulation tool presented the same results, besides the fact that interviews with the dwellers validated the possible flaws that the system could be presenting in the field. As future work, the proposed method will be extended to start from a list of commercial equipment, where each equipment is

verified and the final solution, which satisfies the project specification, is found, thus leading to an optimum sizing of PV systems. We will also consider other types of renewable energy and even hybrid ones to allow our method to design and verify typical rural electrification.

Acknowledgments

The authors would like to thank Coventry University for the availability of the case studies (real PV systems) to validate the tool and to support the mobility of one researcher to UK.

Funding

This work was supported by Newton Fund [grant number 261881580] with the mobility of one researcher to UK; FAPEAM - Amazonas State Foundation for Research Support [grants from call 009/2017 and PROTI Pesquisa 2018], for the financial support related to the Virtual Machine rent (December 2018 to January 2019), for the DCTIEX II scholarship (November 2018), and for support the mobility of the researcher; and FAS - Sustainable Amazonas Foundation for the PhD scholarship (from November 2017 to March 2019).

[1] M. Hussein, 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, 2012, pp. 2656–2666.

[2] IEA International Energy Agency, World Energy Outlook 2018, IEA, Paris, 2018.

[3] Empresa de Pesquisa Energtica EPE, Sistemas Isolados - Planejamento Ciclo 2018 - 2023, http://www.epe.gov.br/sites-pt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-346/EPE-NT-Planejamento- 2018 [Acessed 04 April 2019].

[4] S. Coelho, A. Sanches-Pereira, L. Tudeschini, J. Escobar, M. Poveda, N. Coluna, A. Collin, E. L. Rovere, A. Trindade, O. Pereira, Biomass residues as electricity generation source in low HD source in regions of Brazil, in: UNESP (Ed.), The XI Latin. Cong. of Elec. Gener. and Transm. CLAGTEE, 2015, pp. 1–8. doi:10.13140/RG.2.1.2747.7208.

[5] EPIA, Global market outlook for photovoltaics 2017-2021, European Phototoltaic Industry Association, Belgium, 2017.

[6] S. Pradhan, S. Singh, M. Choudhury, 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. 4 (4) (2015) 203–207.

[7] N. Swarnkar, L. Gidwani, 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. doi:10.1109/ICEETS.2016.7582899.

[8] A. Dobos, PVWatts Version 5 Manual, Tech. rep., National Renewable Energy Laboratory, Colorado (2014).

[9] N. Blair, A. Dobos, J. Freeman, T. Neises, M. Wagner, System Advisor Model, SAM 2014.1.14: General Description, Tech. rep., National Renewable Energy Laboratory, Colorado (2014).

[10] A. Mills, S. Al-Hallaj, Simulation of hydrogen-based hybrid systems using Hybrid2, Int. J. of Hydrog. Energy 29 (10) (2004) 991–999. doi:10.1016/j.ijhydene.2004.01.004.

[11] J. Gow, C. Manning, Development of a photovoltaic array model for use in power-electronics simulation studies, in: Proceedings of the 14th IEE Electric Power Applications Conference, Vol. 146, 1999, pp. 193–200. doi:10.1049/ip-epa:19990116.

[12] A. Benatiallah, D. Benatiallah, T. Ghaitaoui, A. Harrouz, S. Mansouri, Modelling and simulation of renewable energy systems in Algeria, Int. J. of Sc. and App. Inf. Tech 7 (1) (2017) 17–22.

[13] E. M. Clarke, T. A. Henzinger, H. Veith, Introduction to model checking, in: Handbook of Model Checking., 2018, pp. 1–26. doi:10.1007/978-3-319-10575-8_1.

[14] W. Akram, M. A. Niazi, A formal specification framework for smart grid components, Complex Adaptive Systems Modeling 6 (1) (2018) 5. doi:10.1186/s40294-018-0057-3.

[15] D. Kroening, M. Tautschnig, CBMC C Bounded Model Checker (competition contribution), in: Tools and Alg. for the Const. and An. of Sys. (TACAS), Vol. LNCS 8413, 2014, pp. 389–391.

[16] M. Gadelha, F. Monteiro, J. Morse, L. Cordeiro, B. Fischer, D. Nicole, ESBMC 5.0: An industrial-strength C model checker, in: 33^{rd} ACM/IEEE Int. Conf. on Aut. Soft. Engin. (ASE'18), ACM, New York, NY, USA, 2018, pp. 888–891. doi:10.1145/3238147.3240481.

[17] E. Natsheh, A. Albarbar, Solar power plant performance evaluation: simulation and experimental validation, in: J. of Physics: Conf. Ser., Vol. 364, 2012. doi:10.1088/1742-6596/364/1/012122.

[18] M. Lamnadi, M. Trihi, A. Boulezhar, Study of a hybrid renewable energy system for a rural school in Tagzirt, Morocco, in: Int. Ren. and Sust. Energ. Conf. (IRSEC), 2017. doi:10.1109/IRSEC.2016.7984079.

[19] Y. Driouich, M. Parente, E. Tronci, Modeling cyber-physical systems for automatic verification, in: 14th Int. Conf. on Synth., Mod., Appl. to Circ. Des. (SMACD 2017), 2017, pp. 1–4. doi:10.1109/SMACD.2017.7981621.

[20] A. Abate, Verification of networks of smart energy systems over the cloud, in: S. Bogomolov, M. Martel, P. Prabhakar (Eds.), Num. Soft. Verif., Vol. LNCS 10152, 2017, pp. 1–14. doi:10.1007/978-3-319-54292-8.

[21] Y. Driouich, M. Parente, 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. doi:10.1109/EESMS.2018.8405826.

[22] P. Mohanty, K. Sharma, M. Gujar, M. Kolhe, A. Azmi, Solar Photovoltaic System Applications, Springer International Publishing, 2016, Ch. PV System Design for Off-Grid Applications, pp. 49–83. doi:10.1007/978-3-319-14663-8.

[23] A. Hansen, P. Sørensen, L. Hansen, H. Bindner, Models for a stand-alone PV system, no. 1219, Forskningscenter Risoe, 2001.

[24] E. Saloux, A. Teyssedou, M. Sorin, Explicit model of photovoltaic panels to determine voltages and currents at the maximum power point, Solar Energy 85 (5) (2011) 713–722. doi:10.1016/j.solener.2010.12.022.

[25] J. Cubas, S. Pindado, F. Sorribes-Palmer, Analytical calculation of photovoltaic systems maximum power point (MPP) based on the operation point, Applied Sciences 7 (9) (2017) 870–884. doi:10.3390/app7090870.

[26] M. Villalva, J. Gazoli, E. Filho, Comprehensive approach to modeling and simulation of photovoltaic arrays, IEEE Trans. on Power Elec. 24 (2009) 1198–1208. doi:10.1109/TPEL.2009.2013862.

[27] R. Ross, Flat-plate photovoltaic array design optimization, in: C. San Diego (Ed.), 14th IEEE Photovoltaic Specialists Conference, 1980, pp. 1126–1132.

[28] J. Copetti, E. Lorenzo, F. Chenlo, A general battery model for PV system simulation, Prog. in Photovoltaics: Res. and App. 1 (4) (1993) 283–292. doi:10.1002/pip.4670010405.

[29] J. Pinho, M. Galdino, Manual de Engenharia para Sistemas Fotovoltaicos, CEPEL CRESESB, Rio de Janeiro/RJ, 2014.

[30] T. Khatib, W. Elmenreich, Optimum availability of standalone photovoltaic power systems for remote housing electrification, Int. Journal of Photoenergy (Article ID 475080) (2014) 5 pages. doi:10.1155/2014/475080.

[31] E. Clarke, W. Klieber, Miloš Nováček, P. Zuliani, Model Checking and the State Explosion Problem, Springer, Berlin, 2012, pp. 1–30. doi:10.1007/978-3-642-35746-6_1.

[32] J. Morse, L. C. Cordeiro, D. A. Nicole, B. Fischer, Model checking LTL properties over ANSI-C programs with bounded traces, Software and System Modeling 14 (1) (2015) 65–81. doi:10.1007/s10270-013-0366-0.

[33] J. Morse, Expressive and efficient bounded model checking of concurrent software, Ph.D. thesis, University of Southampton (2015).

[34] P. Schrammel, D. Kroening, M. Brain, R. Martins, T. Teige, T. Bienmüller, Incremental bounded model checking for embedded software, Formal Asp. Comput. 29 (5) (2017) 911–931. doi:10.1007/s00165-017-0419-1.

[35] D. Beyer, M. E. Keremoglu, CPAchecker: A tool for configurable software verification, in: G. Gopalakrishnan, S. Qadeer (Eds.), Lecture Notes in Computer Science, Vol. LNCS 6806, Springer, Berlin, Heidelberg, 2011, pp. 184–190. doi:10.1007/978-3-642-22110-1_16.

[36] International Organization for Standardization, ISO/IEC 9899:2018 Information Technology – Programming Languages – C, https://www.iso.org/standard/74528.html - 2018 [Acessed 14 November 2018].

[37] Weatherbase, Manaus, Amazonas Travel Weather Averages, https://www.weatherbase.com/ - 2018 [Acessed 09 July 2018].

[38] EnergyPlus, Weather data by location, https://goo.gl/A82Jqh - 2018 [Acessed 09 July 2018].

[39] R. Brummayer, 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. doi:10.1007/978-3-642-00768-2_16.

[40] L. D. Moura, N. Bjørner, Z3: An Efficient SMT Solver, in: Tools and Alg. for the Const. and An. of Sys. (TACAS), Vol. LNCS 4963, 2008, pp. 337–340.

[41] A. Cimatti, A. Griggio, B. Schaafsma, R. Sebastiani, The MathSAT5 SMT Solver, in: N. Piterman, S. Smolka (Eds.), Proceedings of TACAS, Vol. 7795 of LNCS, Springer, 2013.