Article

The Branch-and-Bound Algorithm in Optimizing Mathematical Programming Models to Achieve Power Grid Observability

Nikolaos P. Theodorakatos [1,*], Rohit Babu [2] and Angelos P. Moschoudis [3]

[1] School of Electrical & Computer Engineering, National Technical University of Athens, 157 80 Athens, Greece
[2] Department of Electrical and Electronics Engineering, Alliance University, Anekal, Bengaluru 562 106, India; rohit.babu@alliance.edu.in
[3] Department of Electrical and Electronics Engineering, University of West Attica, 122 44 Athens, Greece; amoschoudis@uniwa.gr
[*] Correspondence: nikos.theo2772002@gmail.com or nikos277@central.ntua.gr

Abstract: Phasor Measurement Units (PMUs) are the backbone of smart grids that are able to measure power system observability in real-time. The deployment of synchronized sensors in power networks opens up the advantage of real-time monitoring of the network state. An optimal number of PMUs must be installed to ensure system observability. For that reason, an objective function is minimized, reflecting the cost of PMU installation around the power grid. As a result, a minimization model is declared where the objective function is defined over an adequate number of constraints on a binary decision variable domain. To achieve maximum network observability, there is a need to find the best number of PMUs and put them in appropriate locations around the power grid. Hence, maximization models are declared in a decision-making way to obtain optimality satisfying a guaranteed stopping and optimality criteria. The best performance metrics are achieved using binary integer, semi-definite, and binary polynomial models to encounter the optimal number of PMUs with suitable PMU positioning sites. All optimization models are implemented with powerful optimization solvers in MATLAB to obtain the global solution point.

Keywords: phasor measurement unit (PMU); optimization; optimal PMU placement (OPP); binary integer programming; nonlinear programming; semidefinite programming; polynomial model; branch-and-bound algorithms; necessary and sufficient conditions for optimality; suboptimality criteria; zero-gap optimality; optimizer routines

check for updates

Citation: Theodorakatos, N.P.; Babu, R.; Moschoudis, A.P. The Branch-and-Bound Algorithm in Optimizing Mathematical Programming Models to Achieve Power Grid Observability. *Axioms* **2023**, *12*, 1040. https://doi.org/10.3390/axioms12111040

Academic Editors: Fevrier Valdez and Rafael Bernardo Carmona Benítez

Received: 12 October 2023
Revised: 29 October 2023
Accepted: 4 November 2023
Published: 8 November 2023

1. Introduction

An electricity power grid network is a complex and dynamic arrangement that must be observed without interruption to provide a continuous power supply to the customers [1,2].

The power network state is determined by the assignment of phasor voltages to nodes, electrical current values, and power values to generators and loads. One preferred method to make the power grid smart is to observe its state in real-time [1,2].

These days, power systems are transformed into smart grids (SGs) [1–3]. To implement a modern power system, improved sensors that will observe the power grid's state must be taken into consideration for installation [1–3].

Such technologically advanced sensors are synchrophasor devices, which are well-known as PMUs [1–3]. Over the past two decades, there has been a lot of investigation of synchronized measurement derived by PMUs in wide-area processors for applications such as observation of the entire power grid and stability, control, and protection tasks involved with a secured condition of the grid. Also, a large number of PMUs are installed in transmission and distribution power grids [1,2].

PMU measures electrical quantities such as phasor voltage at a power network node and the phasor current emanating from that bus to the adjacent neighborhood nodes [1].

PMUs are smart devices that provide unprecedented factors leading to success for the operation of the system and control applications due to their accessible voltage phasors and currents related to a frequent global time reference, despite the fact that they are calculated by remote system parts [1].

A PMU is a monitoring device, also known as a synchrophasor, that is installed on a power grid bus. The synchrophasor is able to measure the voltage magnitude and phase angle fully synchronized with a signal achieved continuously from the global positioning system (GPS) [1].

PMUs supply precisely the power quality view around the wide of an electricity power grid [1]. Each synchrophasor device is installed at a selected network bus, collects the voltage at that node and the current measurement towards the neighborhood nodes, and sends them to the system operator for wide-area applications [1].

All electrical quantities are sent to the phasor data concentrator (PDC), which works together with the supervisory control and data acquisition (SCADA) system [1]. Smart grids, together with the Internet of Things, are taken into account as the future of modern electricity power grids [3].

The electricity power grid is defined as fully observable on the condition that the phasor voltage of all system nodes is calculated utilizing synchronized or pseudo-measurements [1,2].

A full observability scenario is defined as the system state where all voltage-phasor buses can be observed. A PMU is a monitoring device that can be installed at a power graph vertex to directly observe the voltage phasor at that vertex and the current phasors of all adjacent edges to that vertex [3–5]. Technical or economic restrictions forbid the installation of these measuring and monitoring devices at each system node [1].

The optimal PMU placement (OPP) problem is an optimization problem [1] abstracted from an approach to observe the state of a power electricity grid. The OPP problem is considered to be a discrete optimization problem reflecting the smallest minimum dominating set problem [4–8]. Its solution is to pose a minimum number of PMUs to observe the whole power grid [9–18].

The power grid is studied as a graph, and for a good enough act of watching over the power grid, the supervision of the power grid needs the monitoring of the voltage phasor at each vertex as well as the current emanating from that vertex to be observed [3–8].

To determine the observability state of a modern power grid, many placement models have been implemented in the past based on finding the optimal number of PMUs [12–18]. An optimization model is based on a constraint function and leads to a solution that satisfies all the restrictions that meet the cost requirement [12–18].

Until now, the OPP problem has been studied to come across a suboptimal or optimal number of PMUs, and a placement site is delivered to cover the complete observation scenario of the power grid state [12–18].

Mathematical or heuristic algorithms encounter the number of PMUs required to be posed at selected buses around the power grid [12–18]. The most commonly utilized algorithm to solve the OPP problem is binary integer programming, and some pioneering works are reported in the bibliography [19–23]. In [24], integer quadratic programming is utilized to handle optimality and measurement redundancy tasks. A Groebner base algorithm is utilized to solve the OPP problem in [25].

Semi-definite programming optimizes a linear objective function under linear matrix inequality (LMI) constraints, whereas the solution is found in a binary decision domain [26–28]. An optimization strategy is presented in which the task of maximum measurement redundancy is considered based on a neighborhood search [29].

A three-phase algorithm is utilized to find the minimum number of PMUs required to render the system entirely observable [29]. The numerical outcome illustrates that the proposed methodology is easy to implement and gives optimal placement sites to achieve effective observation in wide area monitoring systems (WAMS). WAMS is a system operator application that is able to observe the entire power grid through an adequate number of PMUs properly installed at appropriate power grid nodes [1,2].

The authors in [30] utilized the graph theory method to find a solution for the OPP and the minimum PMU number. A genetic algorithm (GA) [31,32], a binary cuckoo optimization algorithm (BCOA) [33], a binary gravitational search algorithm (BGSA) [34], a binary particle swarm algorithm (BPSO) [35–37], and a crow search algorithm (CSA) [38] are utilized to solve heuristically the OPP problem.

GAs [31,32] or BPSO [35–37] heuristically search the feasible areas constituted by the integer linear program's constraint to converge to a local solution point. To reach a desired solution, a comparative study with BBA's metrics is required [32].

This is a general remark that an evolutionary algorithm's output must be compared with that found by BBA to judge if this outcome is the local solution or the global one [9].

In an evolutionary code, there is no sufficient stopping criterion that absolutely indicates when the output is exactly perfect. This makes it impossible to predict when the output will be accurate [9].

Works in [39,40] propose a multi-objective function to be optimized for achieving maximum observability. Both works utilize evolutionary algorithms such as the whale optimization algorithm (MWOA) [39] and the dingo optimizer [40].

The outcomes are credible, and they can be used for validation of the proposed algorithmic scheme's results. Both algorithms are based on a trial-and-error effort to locate a solution with uncertainties about whether this solution is global or not [9–11].

Despite this algorithmic truth, the authors in both works succeed in finding solutions that satisfy the maximum network observability [22]. A two-step method and a quadratic programming method are utilized in [41] to gather those solutions with maximum measurement redundancy. Although there is no strictly mathematical rule leads to a global solution, these algorithms locate the optimal number of PMUs [19–46].

A weighted least squares algorithm is studied where a quadratic function is optimized under an equality constraint function [43]. Although it is promising to use a nonlinear approach in a binary domain, this solution suffers from approximate solutions [43].

A nonlinear optimization algorithm's return must satisfy specific tolerances such as objective function evaluation, constraint violation, and the tolerance of the step size on the decision variable change per iteration [44].

Mathematical algorithms such as recursive quadratic programming (RQP) [45] or generalized pattern search algorithm (GPSA) [46] utilize function evaluations and step-size on the decision variables to end the iterative process with care that the final output is a local and feasible solution at the same time as pointed out in [44].

RQP [45] and GPSA [46] were studied by the authors to solve a nonlinear system where the equality constraints are fewer than the number of decision problems [9]. RQP [45] and GPSA [46] handle the nonlinear problem and result in a pure binary local point without approximate uncertainties on the quality of the solution [45].

Both nonlinear algorithms find locally optimal solutions satisfying specific feasibility and iteration-stopping criteria [44].

Moreover, GPSA is considered an improvement because it optimizes a one-product cost function that leads to solutions with maximum network observability [22]. In this study, binary variables are employed to convert the nonlinear problem originally defined within a continuous variable domain [45,46] into a binary quadratic polynomial problem.

GPSA locates optimal solutions with the same quantity and quality as those found in [34,35,39–41]. Works in [34,35,39–41] optimize a multi-objective function to find a desired solution, whereas a GPSA [46] minimizes a one-cost function to be optimized. All scientific works define the upper limit of the system redundancy observability index (SORI) as proposed in [22], and the outcomes are used as a comparative study to validate our algorithmic scheme results.

These solutions are utilized in this study for further validation of the robustness of the presented optimization results from the aspect of maximum network observation of the power grid [22]. Also, a wide-area monitoring application is studied in [47] to upgrade protection and control issues using PMUs in addition to a communication infrastructure (CI) to bring all PMUs to the PDC [1].

A symmetry topological observability based on the ILP method is proposed in [48] for the solution to the OPP problem. Also, an optimization based on the OPP model is presented in [49] under several contingencies for reliable power system monitoring.

An ILP-based method deliver the number of the intelligent monitoring devices and their best optimal sites in [50].

Nonlinear approaches like RQP [45,51] are reported in [52] for power system and analysis applications. Also, OpenEdgePMU architecture is presented in [53]. Other techniques, such as an optimized extreme learning machine-based procedure and the utilization of synchrophasors to guarantee real-time power transient stability prediction, are proposed in [54].

Lately, micro-PMUs have been installed in power distribution systems, and papers have been published to review the utilization of these monitoring devices in this domain [55–57]. Also, new state estimation tools are proposed in [58,59] or further processing of PMUs and SCADA measurements to obtain better power grid monitoring [1,2].

Also, recently published papers have illustrated the benefit of the utilization of PMU data with the Kalman filter for actual-time state estimation of the power grid [60,61].

Also, a state estimator based on least-trimmed squares is proposed in [62]. A newly published work proposes a state estimation process using SCADA and PMU in AC/DC grids in [63].

2. Knowledge Gap of Past Studies and Contribution Declared in This Work

Algorithms such as GA [31,32], BCOA [33], BGSA [34], BPSO [35–37], CSA [38], RQP [45], and GPSA [46] locate locally optimal solutions either in the continuous or binary decision variable domain. Also, previous efforts in the BBA [20–23] and semi-definite [26–28] domains didn't give the appropriate attention to this optimality criterion.

Integer programming and semi-definite programs are introduced in previous studies [20–23,26] without illustrating that the solution does not suffer from uncertainties related to suboptimality criteria. Moreover, the proposal of SDP in [28] suffers from uncertainties if the return solution satisfies the maximum network observability [22].

None of the cited articles dealing with binary or continuous decision variables in the OPP gives prominence to the issue of solution uncertainty. Optimization models such as binary integer linear programs [19–23], 0/1 semi-definite programming formulation [26–28], and quadratic binary polynomial problems are solved by a BBA implemented by the YALMIP software package (https://yalmip.github.io/ accessed on 15 October 2023) [26–28].

In addition to solving such kinds of optimization problems through BBA, a solution must fulfill a suboptimality criterion [9–11]. As far as we know, there is not much attention to deal with a solution if it ranges in the 0.00% optimality criterion [20–23].

This study resolves the OPP problem and presents a branch-and-bound algorithm for a binary integer program [19–23], a Boolean semi-definite program, and a binary quadratic polynomial model [26–28].

This work studies these optimization problems and solves them exactly by the BBA in a zero-gap tolerance, and based on this computational logic, the maximization problem is also considered, studied, and solved [9–11].

This optimality criterion is primarily utilized to find any solution for either the minimization or maximization problem in fields such as binary integer programming [19–23], semi-definite programs [26–28], and polynomial models. All optimization problems are written in YALMIP using symbolic syntax, with a strict declaration of the binary nature of the decision variables [26–28]. We initially optimize the 0/1 ILP model with a BBA [64–68] implemented by the bmibnb built-in solver in the YALMIP program in collaboration with external optimizer routines to find optimality [69–79].

It invokes an ILP routine to estimate the first feasible solution, the upper and lower bounds, and finally to obtain a global solution point [70,72–74,77,79]. This optimization solver utilizes a standard BBA, builds a binary tree, and constantly calculates an upper and lower bound estimation on the objective function, whereas a percentage relative gap is given at each explored node [70,72–74,77,79]. Finally, an optimal solution has been discovered with a 0.00% optimality criterion [9–11,42,44].

To solve the quadratic constraint polynomial problem, the bmibnb built-in YALMIP solver invokes two optimizer routines, such as a local NLP solver [70] and an ILP solver [69–74,77,79].

The bmibnb invokes the MATLAB® function fmincon to find the first feasible solution and an upper bound on the objective function [71]. On the other hand, the MATLAB® function intlinprog [70], for example, is utilized to solve the linear programming relaxations at each explored node and estimate the lower bound [70].

The optimization routine constantly estimates the difference between those bounds and delivers an optimal solution within a 0.00% criterion [9–11,42,44,76,77]. Therefore, no better solution can be found, and this point is a global one [76].

Finally, the CUTSDP built-in YALMIP solver relaxes the conic constraints and solves the Boolean semi-definite program either in the minimization or maximization problem [69,78]. It also invokes any powerful MILP solver to count the optimality criteria through the cutting-plane strategy in optimizing the mathematical model [70,72,74,79]. The entire optimization process results in a global optimum point [9–11,42,44].

3. Novelty and Computational Methods Related to Discrete Optimization

The goal of this work is to improve the reliability of the local and wide transmission grids by using PMUs to enable and enhance system monitoring, control, and protection applications [1]. As a result, the power system state is obtainable in actual time [1].

As the PMU installation in a power grid transmission system is yet a costly devoting investment, it is not possible to pose PMU at each power grid node aiming at power grid monitoring. As a result, the motivation is to find global algorithms to find out the exact number of PMUs in a single run without needing any comparison with the existing methods and where this algorithm is superior to the previous algorithm.

Our algorithmic scheme can find the accurate number of PMUs and their location sites within an adequate stopping criterion that satisfies globality [9–11,42,44].

The main algorithm is the BBA, which solves the binary integer program, the semi-definite program, and the quadratic constraint Boolean model.

Another target is to declare a maximization problem to obtain those solutions to ensure the maximum observation of the power grid monitoring, which is a challenging task.

The aim of the system operator is to constantly supply electricity to the customers. To implement a combinatorial problem as the OPP, a decision computational way must be followed. The objective function is stated generally as [11]:

$$\min J(\mathrm{x}) == \sum_{\mathrm{i}}^{\mathrm{n}} w_i | \mathrm{x_i} = true/false \tag{1}$$

Also, a logical constraint function is taken into account to cover the complete observation scenario of the power grid state as first declared in [11,20,21].

$$F(\mathrm{x}) = F(\mathrm{x_1}, \mathrm{x_2}, \ldots, \mathrm{x_n}) = \begin{cases} true \\ false \end{cases} \tag{2}$$

This optimization formulation model is stated in an $go - no - go$ integer linear program defined on decision variables with a binary nature. In this process, the decision variables are stated to be $0 - 1$ values [9–11]. As a result, an assignment problem is declared [10]. Branch-and-bound algorithms (BBAs) are capable of solving assignment problems, such as the OPP, or power dominating set in graph theory terminology [5,19].

1. Minimizing the number of PMUs achieves optimization of the models towards a solution with a 0.00% optimality criterion. Consequently, the placement of PMUs at appropriate sites reduces the total installation cost.
2. Proposing maximization models that yield solutions with maximum observatory indicators. In other words, the reliability of the power grid is enhanced by achieving the upper limit related system indicator, by which a power grid network node how much time is observed.
3. So, if solutions with maximum SORI are placed at appropriate sites, the electricity transmission system becomes better at monitoring and avoiding interrupted conditions. This index defines the total knowledge of the observations for the WAMS.

Our algorithmic approach is based on four parameters to obtain an optimal solution inside a zero absolute gap and a percentage relative gap equal to 0.00% [76]:

(1) To obtain an adequate number of PMUs by minimizing a linear objective function under a set of topologic or numerical constraints, the topologic constraints are formulated considering Kirchhoff's and Ohm's rules [20–23]. On the other hand, the numerical constraints are based on a linear estimator procedure [26–28].

(2) To achieve those optimal PMU arrangements where each power grid node is observed for a maximum time, directly or indirectly [22,34,35,39,41–46].

(3) As a result, minimization and maximization problems are symbolically programmed in MATLAB using a specialized library like YALMIP [69].

(4) The minimization problems are solved by a BBA building an enumeration tree to find a global solution point inside an objective function gap [9–11,44]. On the other hand, the maximization problems are declared based on the optimal solution returned by the minimization model. Then, the BBA returns optimal solutions with high quality, satisfying the maximum network reliability for each mathematical model.

4. Power Dominating Set (PDS)-Optimal PMU Placement Problem

Many integer programming (IP) problems arise in graph theory. Graph theory and integer linear problems have been extensively treated elsewhere, for example, by Christofides (1975) [8]. One of the most significant relevances of graph theory in mathematical computing science is its utilization in electrical network theory [3–8].

Let $G = (V, E)$ be a graph reflecting an electric power network, where a vertex depicts an electrical node (a substation bus where transmission lines, loads, and generators are connected) and an edge depicts a transmission line connecting two electrical nodes.

The optimal PMU allocation maps to the smallest minimal dominant set on the graph. The OPP problem is also known as the Power Dominating Set (PDS), which needs a minimum number of PMUs to pose on the nodes. As the graph is fully observed, the network state is to be calculated [3–5].

Haynes et al. considered the graph-theoretical representation of this problem as a variation of the power domination set problem [5]. They declared a set S to be a power-dominating set of a graph if every vertex and every edge in the power system is observed by the set S that follows a set of rules for power network observation [5,19].

As a result, the problem of the installation of the smallest set of PMUs to observe the whole power system is a graph theory problem physically closely related to the commonly accepted vertex covering and domination problems [3–5,19].

A graph $G = \{V, E\}$ consists of objects $V = \{v_1, v_2, \ldots\}$ called vertices and another set $E = \{e_1, e_2, \ldots\}$, whose elements are called edges, such that each edge e_k is recognized with an unordered pair $\{v_i, v_j\}$ of vertices. The vertices $\{v_i, v_j\}$ associated with the edge e_k are called the end vertices of e_k [6–8].

A PMU calculates the voltage and phase angle for the vertex at which it is installed, its adjacent edges, and their end vertices. These vertices and edges are considered to be monitored. Any vertex that is adjacent to a monitored edge is observed, and any edge joining two monitored vertices is observed [3–5].

The problem of observing an electricity power network is solved by placing a selected number of measurement devices around the power system, which is closely related to the well-known domination problem in graphs [1,5,19].

A PMU calculates the state variable, which is the voltage and phase angle for the vertex at which it is installed and its adjacent edges and their end vertices [1].

Definition 1. *Declaration of the observability laws related to the PDS model [5,19]:*

1. *Any vertex that is adjacent to a monitored edge is observed.*
2. *Any edge connecting two observed vertices is monitored.*
3. *If a vertex is adjacent to a total of $k > 1$ edges, and if $k - 1$ of these edges are monitored, then all k of these edges are observed.*

The binary connectivity matrix, A_{PMU} as declared in [20–23], is given by the "vertex-node adjacency matrix" for the matrix $X = [\mathrm{x_{ij}}]$, on which the elements along the main diagonal of it are adjusted to 1, that is $[\mathrm{x_{ij}}] = 1, \forall \mathrm{i} = \mathrm{j}$ [6–8].

Provided that all the diagonal elements of the matrix are 1 on condition that every vertex is "reachable" from itself by a path of cardinality 0, the set of $\mathrm{x_i}$ which are reachable by $\mathrm{x_j}$ along a path of cardinality 1, i.e., the set of the arcs $(\mathrm{x_i}, \mathrm{x_j})$ exist in the graph, as noted in [6–8]. As a result, the binary connectivity matrix as declared for solving the OPP problem is the reachability matrix $R = [\mathrm{r_{ij}}]$ as defined in [5–8]:

$$r_{\mathrm{ij}} = \begin{cases} 1\ if\ vertex\ \mathrm{x_j}\ is\ reachable\ from\ vertex\ \mathrm{x_i} \\ 0\ otherwise \end{cases} \tag{3}$$

Evidently, all the diagonal elements of R are 1 since every vertex is reachable from itself by a path of cardinality 0 [5–8]:

$$A_{k,m} = \begin{cases} 1, \text{ if } k = m, \text{ or } k \text{ and } m \text{ are connected} \\ 0, \text{ otherwise} \end{cases} \tag{4}$$

The determination of the minimum cardinality power dominating set is equivalent to the PMU placement methodologies that can completely monitor the network state, whereas the system operator utilizes a minimum number of synchronized sensors to be installed at selected power grid nodes.

We utilize basic circuit rules, that is, Kirchhoff's Voltage Laws and Ohm's Law, and measurement sets directly achieved by PMUs, which aim at determining the voltage phasors at all power grid nodes that are not directly measured [19,23,32,45,46].

The first observable rule of optimal allocation of PMUs is: A bus that has a PMU installed at that bus will have its voltage phasor and all branch currents adjacent to it observed by the PMU monitoring device [19–21].

The second one is as follows: By applying Ohm's law, the voltage phasor at one end of a branch current can be computed if the voltage phasor at the other end of the branch current is known [19–21]. Through the simulation results, it is proven that there is a direct connection between the optimal allocation of PMUs and the determination of the smallest dominating set. The resulting optimization model being constructed for that purpose is to find the minimum number of PMUs that are adequate to observe the electricity power transmission grid, which is also mentioned as the OPP problem.

The OPP problem is declared based on Kirchhoff's Voltage laws and Ohm's Law and was at the beginning presented by Abur [20,21]. The work by Haynes et al. [5] was one of the first of this kind to utilize a graph theoretical approach for solving the PDS as an OPP problem. Based on graph theory, this study utilizes the Vertex-to-Vertex Connectivity Matrix to build the binary connectivity matrix to reconsider the binary integer program as presented in [20,21]. The steps of the implementation of the Vertex-to-Vertex connection matrix are the following in the optimization run of the integer linear program [19–23].

The pseudo-code presents the Algorithm 1 used to build the Vertex-to-Vertex Connectivity Matrix [6–8] in Algorithm 1.

Algorithm 1: Vertex-to-Vertex Connectivity Matrix

```
Outcome: Vertex-to-Vertex Connectivity Matrix
Variable: int $i$, $k$, $n$; Matrix A;
for $i \leftarrow$ 1 to $n$ do
    for $k \leftarrow$ 1 to $n$ do
        if $i = k$ then
            A [$i$, $k$] $\leftarrow$ 1;
        else
            if $i$ and $k$ are connected then
                A [$i$, $k$] $\leftarrow$ 1;
            else
                A [$i$, $k$] $\leftarrow$ 0;
            end
        end
    end
end
```

A graph vertex is considered to be directly monitored if a PMU is posed at that vertex and also indirectly monitored by a PMU installed at its incident vertex [5].

If all vertices in the graph are observed directly or indirectly, the graph is considered to be fully observable [5]. The number of lines connecting with each vertex aims at maximizing the maximum observability indicator so that the measurement redundancy can be preserved [22,33–35,39,41,46].

The neighborhood buses connected to a network bus with a PMU help a lot to count the BOI, the summation of this index reflecting the SORI, and a global solution point [22].

The resulting solutions have such BOI indices, counting the maximum SORI satisfying a secure state estimator process [22,33–35,39,41,46].

During the first phase, the optimization of the binary program is conducted, assigning equal weight to each decision variable. Analyzing the format of the binary program is performed based on the connectivity of the transmission lines, followed by an exact solution using a BBA [9–11].

Subsequently, a revisitation of the semi-definite program is performed, leading to the declaration of a quadratic constraint binary program [9–11].

All minimization problems are solved by a BBA in an absolute gap equal to a zero-value price. In the second stage, a maximization problem is declared, incorporating an additional linear constraint that reflects the optimal solution achieved in the minimization problem.

5. Mathematical Programming Models

The study of PDS in a graph representing an electricity transmission system network aims at placing a minimum number of PMUs to observe the entire grid state [4].

It is illustrated that the PDS is NP-hard to optimize, and this remark is based on the bibliography [3–5], but it can be computed exactly without using evolutionary algorithms to locate a suboptimal solution [12]. The theoretic analysis is given by previous studies in the graph theory domain [3–5].

In fact, the set of power nodes where the PMUs have to be posed reflects a dominant set of the graph [3–5]. As a result, the OPP problem maps to the smallest dominating set problem on the graph, as pointed out in [3–5].

The combination of the PDS with the OPP problem has been adopted. As a result, revisiting certain programming models is necessary to solve the OPP problem exactly.

Our purpose is to give a computational way to solve it through reconsidering the existing models and also to propose a quadratic constraint binary problem. The integer programming model is the state-of-the-art computational method to solve the PDS or OPP, showing competitive elapsed time performance metrics [3–5].

This work presents a mathematical formulation and implementation in YALMIP in collaboration with MATLAB to find a cure for the determination of a global solution point related to minimization or maximization problems [69,77].

The BBA built-in BMIBNB function, which is global nonlinear nonconvex integer programming, is utilized for solving binary integer and polynomial problems and a CUTSDP optimizer routine to the 0/1 SDP problem solving [69–74,77]. The proposed algorithmic scheme is applied to validate the functionality based on two observability scenarios [20–28,32,45,46]:

I. Complete Observability
II. Maximum Network Observability

The first scenario is to ensure that the power grid state is fully observed, whereas the second scenario gives those solutions to the state estimator tool for maximum reliability of the measurements.

The algorithm is the BBA to solve it, and the same algorithm is followed to solve the Boolean semi-definite program and the quadratic constraint programming model.

Although the PDS or OPP is known to be NP-complete [4], the instances are presented in this study, and they are solved exactly within a zero-valued absolute gap.

The PDS has been widely studied in the literature [12–18]; for example, it is an NP-complete problem [4,5,19]. Despite this general truth, the instances studied in this work are exactly solved by BBA [9–11,70–72,74,77].

BBA can solve the proposed models by building an enumeration tree, exploring nodes, solving linear programming relaxations, and finally getting a solution with a 0.00% optimality criterion [77].

This criterion means that no higher-quality solution can be found; hence, the optimization problem can be solved globally [77]. The YALMIP state-of-the-art mathematical models are used for integer, semi-definite, and polynomial problems [77].

BBA solves the binary integer and polynomial problems following the same tactic procedure. An upper solver estimates the upper bound on the objective value, whereas a lower solver produces the lower bound to obtain the best objective bound [77].

Also, an LP-optimization function is utilized to explore the nodes, solve linear problems, and implement a suitable branch strategy to build the BBA tree [9–11].

Finally, a global solution point is achieved for both optimization models that range in the binary domain. We also utilize the other built-in YALMIP solver called CUTSDP to relax the semi-definite program's conic constraint function, where a linear objective function is optimized under conic constraints in a binary symbolic format [26–28,78].

This function calls a powerful advanced ILP solver to return an optimal solution for the minimization and maximization problems [70,72,74,79]. The innovation of this work entails the fact that optimization problems are symbolically programmed in MATLAB [10]. The optimization model is written using a specialized YALMIP library, which derives a log file that sufficiently shows that a global solution point is derived within an absolute and relative gap both equal to zero [69–74,77].

5.1. Constrained Integer Programming with Binary Decision Values

Most practical integer program models restrict the integer variables to two values; such 0/1 variables are utilized to represent "yes" or "no" decisions [10].

The electricity transmission grid is fully observable if the voltage phasor is computed for each power bus directly or indirectly [19–21]. The most frequent conception formulation is a constraint integer program with binary decision variables [20–23].

The PDS problem can be interpreted through a constraint-integer linear programming model with binary-valued variables [5].

Logical associations between such decisions can often be enforced using linear constraints. As the assignment placement variables in the OPP problem demand a value of only 0 or 1, BBA locates the decision variable vector in that domain [9–11].

To obtain a complete observation of the power grid state using the mathematical models, we found the exact number of PMUs replying to their localization sites around any IEEE power transmission network [80].

As a result, the OPP is initially written in an integer program with a binary decision domain. Let n be the number of decisions to be charged with a task. For an n-node electrical power grid, the optimization model is stated as follows in integer linear programming with binary decision variables [20–23].

Definition 2. *Declaration of the nature of the decision variables in decision-making and problem solving [20–23]:*

The decision variables are set to be as follows [20–23]:

$$x_i = \begin{cases} 1 \text{ if a PMU is installed at bus i} \\ \quad 0 \text{ otherwise} \end{cases} \tag{5}$$

The 0/1 ILP model is written as follows [20–23]:

$$\min \textstyle\sum_{i=1}^{n} x_i \tag{6}$$

$$\text{s.t. } A \cdot \overrightarrow{x} \geq \hat{1}$$

Let us define the binary connectivity matrix of the power network as follows [20]:

$$a_{ij} = \begin{cases} 1 \text{ if } i = j \text{ or } j \in \mathcal{A}_i \\ \quad 0 \text{ otherwise} \end{cases} \tag{7}$$

The inequality constraints are multiple-choice restrictions or logical constraints [10]. Given that we have the choice of investment to be at least one of the available alternatives, these constraints have a Boolean logic [10].

Where $x = (x_1, x_{2,}, \ldots, x_n)^T$ is a binary design variable vector, whose entries are declared in the current yes or no decision [20–23].

The objective function (6) reflects the cost of the entire investment. Constraints involved Equation (7) remain in force for each placement variable, and the objective function is optimized under this constraint function in a binary domain. The MILP model with binary decision variables is solved by BBA for each IEEE power system [80].

As a result, the focus is on bringing into clear view the BBA, which is used to solve linear programming relaxations in order to obtain bounds on the cost function value of the initial model within a binary domain [9–11].

A global solution point is obtainable within a zero MIPGapAb [69–74]. This solution is considered an extra constraint function in the declaration of the maximization problem in a mixed-integer-linear program (Equations (14)–(18)). We highlight the following five elements to solve the 0/1 ILP mathematical model as shown in Algorithm 2 [9]:

Algorithm 2: Steps of BBA implementation

1. Lower bounding methodology: A strategy is utilized to attain the lower bound on the objective function to obtain a solution to a specified sub-problem.
2. Upper bounding methodology: An upper bound is estimated at the optimum solution point.
3. Branching methodology: A process is followed to partition a sub-problem to generate more than two children.
4. Search Strategy: A process is followed to determine the search order to build the binary tree.
5. If the absolute gap between the upper and lower bounds is within a predefined tolerance, the algorithm stops giving an output.

This optimization process can be interpreted as a repetitive scheme to improve the difference between the current upper bound, which is the incumbent objective function value, and the current lower bound, which is the least value of the lower bounds generated by solving the candidate sub-problems as shown in Algorithm 2 [9–11].

The difference between the upper and lower bounds is the optimality gap utilized to achieve termination of the BBA [9–11]. To execute this optimization, MILP solvers such as MATLAB function intlinprog [70], SCIP optimizer [72], Gurobi function and MOSEK [79] can be invoked in the MATLAB environment to obtain a global solution point inside a predefined absolute gap equal to zero [74].

5.2. *Boolean Semi-Definite Programming Model*

Additionally to the constraint integer program, OPP can be formulated as mixed-integer semi-definite programming (MISDP) under linear matrix inequality (LMI) constraints, as presented in [26–28].

0/1 Semi definite programming model considering the OPP problem can be formulated as follows (Equations (8)–(10)), as found in [26–28].

$$\min \sum_{i=1}^{n} x_i \tag{8}$$

$$G(x) = G_0 + \sum_{i=1}^{n} x_i \cdot G_i \succ 0 \tag{9}$$

$$\vec{x} \in \{0,1\}^n \tag{10}$$

The LMI constraint is positive-definite and ensures numerically the power system's observability [26–28]. G_0 is a gain matrix related to the existing power flow and injection measurements in the power network.

On the other hand, G_i is a gain matrix representing the phasor voltage as well as current phasors calculated by a PMU located at a bus i [26–28].

By transforming the inequality constraint of Equation (6) into an equivalent LMI restriction, Boolean semidefinite programming can be stated and symbolically programmed in YALMIP [69]. The main optimizer, the CUTSDP solver, relaxes the LMI constraints [69].

This solver implements a cutting-plane approximation to solve semi-definite programs. With this computational method, the optimization solver relaxes the conic constraint function, solves a mixed-integer-linear program, puts in linear cuts to the mathematical model, and repeats the optimization process until it finds a global solution [69].

To accomplish this optimization and locate an exact optimum point, the program's user invokes an external MILP solver, such as the MATLAB function intlinprog [70], the SCIP optimizer [72], and the Gurobi function [74].

State-of-the-art-optimizer routines such as MATLAB® function intlinprog [70], Gurobi optimizer engine [74], SCIP optimizer routine [72], and MOSEK [79] are utilized to detect those optimal solutions that optimize the linear objective satisfying the LMI restrictions under the Boolean decision variable restriction [26–28].

As a result, the CUTSDP solver [78] is based on a fast MILP solver embedded in a specialized optimization toolbox such as the MATLAB optimization library [70], the Gurobi toolbox [74], MOSEK [79], or an open-source SCIP optimizer routine [72].

5.3. Quadratic Constrained Binary Optimization Model

If the optimization problem is convex, this is adequate to ensure that the solution is at a globally optimal point. Nonconvex problems may have many various local minima points, and selecting the best one is a very hard issue [9–11]. Multi-start procedures are able to run nonlinear problems with high probability, but in real optimization, there is no warranty that the localized point will be a global optimum point [9–11].

One of the models that permits us to declare OPP is quadratic constrained binary optimization (QCBO), which extends quadratic unconstrained binary optimization (QUBO) [68]. QUBO has been used to solve optimization problems such as vertex cover, max-cut, and other graph theory problems [68]. Although QUBOs are NP-complete problems, solutions can be found by utilizing evolutionary algorithms [68].

The proposed QCBO model is an extended nonlinear problem that is symbolically programmed in YALMIP with its BBA built-in solver [77]. A QCBO model is formulated and solved to obtain global optimality [9–11]. The structure of this kind of discrete optimization model is non-convex, with a strict declaration of binary decision variables.

The solution can be exactly found given that the absolute and relative gaps are equal to zero [77]. The QCBO model is declared as follows (Equations (11)–(13)):

$$\min x^T \cdot Q \cdot x, Q \in I_{n \times n} \tag{11}$$

$$f_i\left(\vec{x}\right) = (1 - x_i) \cdot \prod_{j \in n} \left(1 - x_j\right) = 0, \forall i \in n \tag{12}$$

$$\vec{x} \in \{0, 1\}^n \tag{13}$$

Non-convex optimization models [41,42] have multiple local solutions, and theoretically as well as through the optimization, it is difficult to prove which global one is [8]. In this study, we reconsider the optimization problem with a non-convex structure declared in a binary symbolic decision domain [69,77].

Therefore, non-convex problems are transformed into Boolean polynomial problems. Global optimization algorithms such as spatial branch-and-bound algorithms (SBBAs) are utilized to find a global solution point for such an optimization problem as shown in the Appendices A and B [9–11].

The leading concept is to try to find a global solution point regarding model-solving, and the entire optimization run is summarized in the following steps [77]:

1. An s-BBA is utilized and applied to an optimization problem to obtain a global solution point. The performance of the BBA depends on the capability of the relaxations of the polynomial constraint function so that convergence can be achieved [9–11].
2. The iterative process ends up with two explored nodes, as we illustrate in the log files in the Appendices C and D. This solution is established on the optimization run of two external solvers at the same time [9–11].
3. Finally, a solution is found within a predefined absolute and relative gap with zero value at the final explored node of the BBA tree [9–11].

The BMIBNB function optimizes the non-convex problem in the binary decision domain based on linear programming relaxations and a convex envelope approximation strategy [69,77].

This function invokes a local non-linear solver to find the first feasible solution, if one exists. Also, it invokes an ILP solver to solve the linear programming relaxation [69,77].

Finally, the nonlinear solver and the integer solver constantly estimate the upper and lower bounds, calculating the difference between them, and if it goes to zero, a global solution has been achieved [69,77].

5.4. Maximizing Programming Models

In the first phase, the OPP problem results in a globally optimal point. This phase determines the solution that satisfies the complete scenario [21,32,45].

In the second phase, we mainly optimize the results attained in the previous phase and produce those solutions that ensure the maximum reliability of the power network [22,33–35,39,41,46].

PMUs must be installed at vertices with a maximum degree of connectivity to maximize coverage [5,50]. The investigation of the OPP problem involves deriving an optimal solution, wherein each power network node is assessed using the Bus Observability Index (BOI) as a measurement index [22].

This index for a network node is a measuring parameter that determines whether the bus is observed directly or indirectly [22]. The summation of BOI is the SORI, and it is more appropriate for the maximum observability network [22,33–35,39,41,46,50]. Those indices are used to estimate the relative excellence of optimal PMU arrangements [50]. The steps in relation to the algorithm for counting the maximum network observability are summarized as follows:

1. An objective function is declared in getting optimized by giving the number of times a power network is observed.
2. The observability of every vertex is measured, and the BOI is calculated for every vertex.
3. Afterward, the sum of all times by which the power network is monitored is summed, and the SORI is derived.
4. An optimal PMU placement set is given as having the maximum SORI.

The factors of the BOI and SORI perform as a measuring quantity, reflecting the redundancy of measurements [50]. The number of measurements obtained from a power node rises with the larger number of lines attached to that node [22].

The number of PMUs needed for full observability is the same for all the power grids. However, the BOI and SORI rely on the selection of the optimal PMU placement sites.

This quantity is the BOI. The idea of BOI utilization guarantees the spreading out of the PMUs within the power system. As a result, the PMUs number is limited with SORI to give the sign of maximum redundancy given for the resulting set [22].

Maximum observation of the power grid is the supplementary objective of a proposed mathematical program that increases the power grid reliability [22,34,35,39–41,46].

One considerable effect of the OPP problem is to obtain the maximum observability of the whole system in the event of any disruption [22,34,35,39–41,46].

As a result, the maximization problem is stated to maximize the observation of the power grid. In this study, we maximize a cost function, looking for cost-effective solutions in various domains such as ILP, SDP, and polynomial problems.

To solve the problem of maximizing the observability indicator, whereas the system observability is preserved, with a minimum number of PMUs, we implement the secondary optimization problem [22,34,35,39–41,46]. We minimize the OPP problem, and a global solution point is achieved. Afterward, we seek out those optimal solutions with the maximum indicator of the power grid state.

A linear objective function is maximized under ILP [20,21], LMI [26–28], polynomial constraints, and a supplementary linear equality constraint reflecting the PMU number in the solution.

Our study is extended to work out maximization problems, considered secondary problems in getting those solutions satisfying the maximum observation of the power grid state. The optimal solution given by the optimization of the primal problem is considered an extra constraint taken into account in the maximization problem.

i. Constrained Integer Programming with Binary Decision Values

The maximization problem is given in an integer linear program (Equations (14)–(18)), whereas an equivalent problem is proposed in a binary polynomial model (Equations (19)–(22)).

$$\max \frac{1}{n} e^{T} \cdot A \cdot x \tag{14}$$

$$\text{Subject to } \sum_{i=1}^{n} x_i = x_{min} \tag{15}$$

$$\sum_{ij} a_{ij} x_i \geq 1 \tag{16}$$

$$a_{ij} = \begin{cases} 1 \text{ if } i = j \ \text{ or } j \in \mathcal{A}_i \\ 0 \text{ otherwise} \end{cases} \tag{17}$$

$$\vec{x} \in \{0, 1\}^{n} \tag{18}$$

ii. Constrained Polynomial Optimization Problem with Binary Decision Variables

$$\max \frac{1}{n} e^{T} \cdot A \cdot x \tag{19}$$

$$f_i\left(\vec{x}\right) = (1 - x_i) \cdot \prod_{j \in n} \left(1 - x_j\right) = 0, \forall i \in n \tag{20}$$

$$\sum_{i=1}^{n} x_i = x_{min} \tag{21}$$

$$\vec{x} \in \{0, 1\}^{n} \tag{22}$$

iii. Boolean Semi-Definite Programming Model

The maximization problem is also declared in a Boolean Semi-definite Programming model in getting that solution with a maximum observability indicator.

$$\max \frac{1}{n} e^{T} \cdot A \cdot x \tag{23}$$

$$G(x) = G_0 + \sum_{i=1}^{n} x_i \cdot G_i \succ 0 \tag{24}$$

$$\sum_{i=1}^{n} x_i = x_{min} \tag{25}$$

$$\vec{x} \in \{0, 1\}^{n} \tag{26}$$

where x_{min} is the minimum number of PMUs achieved by the primary-based OPP for full observability. The optimization framework is summarized as follows:

1. The BILP model is written in symbolic format, where the YALMIP BBA solves the problem by invoking a powerful advanced ILP solver to count the upper and lower bounds and solve the linear programming relaxations.
2. The Boolean polynomial model is transformed into a polyhedron and solved by the YALMIP BBA function, which constructs a BBA tree where the upper and lower bounds are computed to be equal at the final root node.
3. The Boolean SDP model is also written in the same programming language, where the CUTSDP relaxes the second-order conic constraints through a cutting-plane strategy implemented by an ILP function.
4. The optimization ends with accuracy because the difference between these bounds delivers a zero-tolerance gap. This tolerance measure means that a global optimal solution is obtained.

5. The objective value is considered to be the upper bound for the minimization problem and the lower bound for the maximization problem.

The binary integer program is symbolically programmed in YALMIP software, where the BBA built-in solver interfaces with external MILP solvers to count the upper and lower bounds and the relaxation problem-solving at the explored nodes [77].

The QCBO model is solved with a polyhedron approximation, building a BBA tree utilizing linearized techniques. The output is obtained within an absolute and relative gap, with values equal to zero [77].

This linearization is performed using a software package that utilizes a standard MILP solver in collaboration with a nonlinear function [9–11]. The YALMIP BBA calls these optimizer routines to count the upper and lower bounds, to implement a branching strategy, and finally to return a global optimal solution [77].

The CUTSDP built-in solver relaxes the SDP's conic constraint function, and a cutting-plane strategy through a MILP solver transforms the model into a MILP model [26–28]. Then, the BBA solves the MILP problem exactly with a cutting-plane strategy [9–11].

The entire optimization process is a required thought task related to the utilization of global optimization functions in 0/1 decision-making problem solving [9–11]. All optimization models are solved either in minimization or maximization format.

The optimality criterion yields an attractive computational method to find the value of the quality of the solution given by the solution of each mathematical programming model. The algorithm returns a solution within a 0.00% optimality metric for either the minimization or the maximization problems [9–11].

iv. Solving the optimization models with a BBA inside an absolute gap

The easiest and most intuitive computational manner to solve a 0/1 combinatorial problem as the OPP, either in minimization or maximization format, is to take into account the 0/1 decision variables as a continuous variable [9–11].

Specifically, the CUTSDP relaxes the conic constraint function, by which a linear objective function is optimized in a symbolic binary domain [69,77]. In the first instance, a cutting-plane strategy transforms the Boolean semi-definite program into a mixed-integer solver, and in the second one, an upper and lower solver are taken into consideration for counting and comparing the upper and lower bounds on the objective function [9–11].

On the other hand, the YALMIP BBA solves the binary integer linear and quadratic polynomial models in the same computational way [69].

The YALMIP BBA is a spatial BBA implementation. The steps of the implementation of the spatial branch-and-bound algorithm (s-BBA) in the YALMIP solver are summarized as follows [8–11,69,76–78]:

1. Initialization;
2. Choose region;
3. Lower bound;
4. Upper bound;
5. Pruning;
6. Check region; and
7. Branching.

The proposed mathematical models are solved by a BBA following strictly mathematical rules, avoiding suboptimal solutions. Solving the linear programming relaxations produces a bound on the optimum point and gives the optimal solution for all optimization models studied in this work. This truth is confirmed by the log files that show convergence at a constraint minimum point, avoiding a suboptimal solution [9–11,42].

The suboptimality criterion is calculated at the final stage of the implementation of BBA and is fully satisfied, which means a globally optimal solution has been found.

The maximization models are stated in a decision-making way to obtain optimality satisfying a stopping criterion under guarantee. The best optimality metrics are reached using binary integer, semi-definite, and binary polynomial models to find out the best number of PMUs with the maximum network observability indicator [22].

We declare the quadratic polynomial optimization problem in a symbolic language in MATLAB, and the decision variables are binary. A global search methodology is required to obtain a global solution. A BBA is implemented by YALMIP software to achieve a solution point with a fully met optimality criterion [77].

BBA solves the Boolean polynomial model by relaxing the polynomial constraint function, the binary resection of the design variables, and constructing an enumeration tree to obtain a global optimal solution. Two optimization processes are used to find optimality.

The first optimization function estimates the upper bound, and the second one computes the lower bound in the s-BBA tree implementation. An LP solver solves the relaxed problems, and a suitable branch strategy is followed [9–11].

The built-in YALMIP solver interfaces the MATLAB® function fmincon to calculate the upper bounds and a powerful advanced ILP function to estimate the lower bounds and solve the relaxed linear programming problems [69].

Those routines measure the difference between those bounds and finally detect a global optimal solution within a zero-gap optimality as well as a percentage relative gap. Let us define the s-BBA implementation in the following stages as shown in Table 1 [77].

Table 1. *Steps of* BBA *implementation*.

1.	A process to work out the lower bounds.
2.	A process to work out the upper bounds.
3.	A branching process and exploring nodes.
4.	A termination to be met satisfying predefined tolerances, such as an absolute zero-gap and a percentage relative gap.

To begin the iterative process, the binary restriction $\vec{x} \in \{0,1\}^n$ in the initial problem is replaced by the following interval-continuous constraint [9–11].

$$0 \leq \vec{x} \leq 1 \tag{27}$$

BBA follows an LP-relaxation problem that is being solved where the binary integer requirement on the decision variables is substituted by the weaker constraint presented in Equation (27). If the solution is an integer, then the algorithm terminates [9–11].

Otherwise, two new sub-problems are created by a branching strategy on a fractional variable. The BBA concept is to end the optimization run the moment that the incumbent solution's quality is evaluated by the measured absolute as well as relative gap to be satisfactory within the prescribed tolerances [9–11,74].

The incumbent solution is always the best solution found up until now in the BBA search tree [9–11]. The termination criterion is adjusted to be equal to the TolGapAbs Gap $= 0$ [72]. The entire optimization process results in a ϵ-optimality criterion. The quality of the numerical results was measured by the relative TolGapAbs [72,73]:

Definition 3. *Definition of the percentage relative to TolGapAbs:*

$$\text{percentage relative gap} = 100 \cdot |\text{primal} - \text{dual}| / \text{MIN}(|\text{dual}|, |\text{primal}|) \tag{28}$$

BBA solves the semi-definite programming [26–28], binary integer [20–23], and polynomial models exactly with an optimal solution and a zero-gap tolerance [9–11,74].

That means that no higher-quality solution can be found, and the solution is not suffering from suboptimality uncertainties [9–11,74]. Therefore, a global solution point is achieved for all instances.

6. Suboptimality Criteria and Convergence Tasks

Global optimization is a well-accepted and widespread application in various domains of science and engineering. A global optimizer solver is used to solve the binary integer as well as the quadratic Boolean constraint minimization problems [9–11].

Also, a semi-definite solver named CUTSD is utilized to relax the conic constraint of the 0/1 semi-definite programs [26–28] with the help of a cutting plane strategy to solve it as a mixed-integer program as the log presents in the Appendices C and D.

The cutting-plane strategy is utilized to tighten the linear programming relaxations in order to obtain the solution. Then, the maximization problems are solved based on the optimal solution achieved by solving the minimization problems in order to obtain the best reliability optimal solution for WAMS applications [1,2].

BBA is the only algorithm able to solve all the instances studied in this work. In this study, we present a BBA convergent for the OPP problem and its models. We consider convergence on three-parameter criteria: (I) upper objective bound, (II) lower objective bound, and (III) tolerance objective gap [9–11].

An algorithm produces an iterative process that is fully convergent when the iteration run goes to the objective function evaluation nearer and nearer to a desired solution. BBA runs the optimization runs and constantly measures the difference between the upper and lower bounds to be less than a default value, which is a specified tolerance.

The optimization solver returns an optimal solution for the objective function gap, and the simulation results validate our symbolic programming models without needing to compare them with the existing ones published in the bibliography [16–18].

When solving each optimization model, the BBA constantly works out both an upper and lower bound on the objective function value. The mixed-integer programming (MILP) solver ends up with an optimal result when the gap between the objective upper and lower bounds is less than MIPGapAb [9–11,74].

More accurately, the incumbent solution is the upper bound for a minimization problem, whereas a lower bound is taken into account in order to count the absolute gap; its acronym is MIPGapAb [9–11,74].

For a minimization model, the objective upper bound is considered the best-known solution and feasible at the same time, whereas the bound of the best possible objective is the lower bound [9–11,42,44]. On the other hand, the lower bound is considered the best-known solution for the maximization problems [9–11,74].

6.1. Termination Criteria of BBA for Convergence of the Optimization Run [64–67]

BBA is an in-front algorithm that is utilized to solve the proposed methods due to a good enough stopping criterion to fully accomplish the optimality criterion. As a result, the integer solver successfully returns a globally optimal point, avoiding being trapped in a local minimum or a suboptimal point [74]. Our aim is accomplished, that is, to solve exactly the minimization and maximization problems [9–11,74].

The quality of the solution depends on the termination criteria used in the implementation of BBA. The termination criteria are the absolute gap and the relative gap. The default tolerances are given by the software specification [77,78].

The absolute gap is declared equal to zero. If the BBA returns a solution with a zero-valued absolute gap, then the solution is characterized as a global one [70,72,73]. BBA ends up after spending a limited number of iterations [9–11,74].

To this end, the optimizer solver considers some stopping criteria in order to give a global optimal solution [44]. BBA embedded in an integer linear programming (ILP) solver such as MATLAB® function intlinprog [70] ends up if the difference between the internally calculated *upper* and *lower* bounds on the objective function is less than or equal to Absolute Gap Tolerance [9–11,74].

Definition 4. *The mixed-integer program's absolute gap (MIPGapAb) is declared as [9–11,69–74,77]:*

$$Gap = upper - lower < \in \tag{29}$$

where $\in$ is the default tuning parameter to return an optimal solution point under warranty [70]. If, for a prescribed tolerance $\varepsilon > 0$, the BBA produces bounds, then the termination criteria described in Equation (29) are fully satisfied [9–11,74].

The optimization function observes this difference. If the algorithm is converging, anticipating that $\varepsilon \to 0$, and the solver terminates the optimization when the tolerances are relatively small [9–11,70–74].

In other words, the algorithm terminates at an optimal solution or ends up at an ε-optimum point based on the stopping criterion tolerance given in Equation (29) [9–11].

This prescribed tolerance is considered to be zero by the MATLAB® function intlinprog or relatively small $1e - 04$, defined by the Gurobi MILP solver [70,74].

Definition 5. *The relative gap is declared as found in [70]:*

$$relative\ gap = (upper - lower)/(1 + |upper|) \tag{30}$$

The optimizer MILP solver calculates, without stopping the upper and lower bounds on the objective function, the difference between those bounds. Absolute and relative gaps are computed and confirm the entire optimization run of the algorithm [9–11].

The BBA's stopping criterion is determined by three states: optimality, elapsed time, and memory utilization. BBA solves all instances rapidly, as the log files show in the Appendices C and D. The tuning parameters help a lot to achieve convergence to a desired outcome within the predefined tolerances [9–11,42,44].

A global solution point is dependent on the satisfaction of parameters such as feasibility, tolerance, and optimality metrics [44]. As illustrated by the log file of the optimizer routine, the lower bound is a culprit in closing the gap, giving a certificate of global optimality [9–11,69–74].

The objective function is to obtain a value equal to the upper bound, which is the appropriate cost price for the minimization models [9–11]. On the other hand, the lower bound is considered to be the best solution for the maximization problems [9–11].

6.2. Convergence Analysis

A solution without suboptimality uncertainty is ensured by carefully examining optimality criteria, optimal solutions, and convergence to an adequate stopping criterion.

The LP-based BBA ensures convergence when a convergent branching method is utilized in the MILP solver implementation. A feasible solution is produced and satisfies the original model constraint function, or an infeasible output is proved or pruned by a bound [9–11]. We illustrate that the BBA converges to a solution by spending a finite number of explored nodes where the linear programming relaxations are solved [9–11].

The optimization function derives an upper bound on the objective function for the minimization models and a lower bound for the maximization models [9–11].

The computation complexity of the s-BBA is based on the fact that it must reach a solution satisfying a specific convergence criterion, such as when $upper - lower < \in$ [9–11,69–79].

The termination criterion is constantly evaluated based on a bound for the possible improvement of some lower bounds in contrast with the upper bounds [9–11].

If the absolute gap between the upper and lower bounds is within a predefined tolerance, the algorithm stops [9–11,74]. The optimizer routine constantly calculates the upper and lower bounds, and the entire optimization run ends up with the best incumbent solution, characterized as a global one [9–11,70,72,74,77–79].

As a result, the convergence is followed by the consideration that the upper $-$ lower is less than the default tolerance [9–11,77]. The algorithm constantly searches for a trade-off relationship between the upper and lower bounds and tries to terminate within an absolute tolerance gap with a predefined gap [9–11].

7. Solution of the Algorithmic Scheme Approaches and Optimization Results

PMU is a costly monitoring device [1]. Each PMU price ranges from USD 1260 to USD 10,000, depending on the number of PMU channel capacities [18].

As a result, it is not practical to install a PMU at every network node of any electricity transmission grid. Cost-efficient solutions are needed, which means that an appropriate number of PMUs must be located around the power grid at selected buses [12–18].

Our goal is to minimize the total cost and also maximize the maximum network observability index of this solution [22,34,35,39,41].

PMU placement algorithmic schemes are basically cost-efficient solutions related to PMU deployment around a modern power grid. They give the system operator the chance to guarantee full condition and improve power grid monitoring.

PMUs provide extensive and precise power monitoring and quality points of view in a large, wide geographic grid. The measurements are collected and help the system operator judge if the voltage, current, and frequency are staying within specific tolerances.

The principal concept of strategically posing PMUs is to help the state estimator tool observe the entire power grid state to be fully observable [2]. In addition to that purpose, our algorithms return solutions with maximum redundancy of the measurements in certain feasibility, tolerance, and optimality metrics [9–11].

The proposed PMU arrangement formulation concludes the terms involved in the objection function to find parameters (I) the number of PMUs, (II) PMU placement sites, and (III) maximize the measurement redundancy [46].

In this, we reconsider 0/1 ILP [20–22], 0/1 SDP [26–28], and quadratic binary polynomial problems to find an optimal solution within a 0.00% optimality criterion [69–73,77]. Many applications simply need a black-box approach that takes a given formulation as an input and delivers an optimal solution as an output. For such black-box procedures, the program's user interacts with the optimizer solver. A command-line interface or an interactive shell calls the optimizer function.

This optimizer function is utilized by invoking the m-script files in MATLAB that contain the description of the objective function to be optimized under the constraint function [10]. We follow a decision-making method of calculation that minimizes an objective function under ILP, LMI, and polynomial constraints [9–11].

BBA optimizes a variety of optimization models, such as binary integer linear program (BILP) [20–23] and Boolean optimization models such as the semi-definite model [26] and the proposed polynomial problem [11].

BBA enacts optimization with multiple-choice constraints on the cost function [9]. BBA has the following merits to solve the ILP [20–23], SDP [26–28], and binary constraint polynomial models. The implementation in MATLAB is as follows [9–11]:

1. BBA reduces the computational complexity based on the constraint function; either binary connectivity, LMI constraints, or polynomial constraints are easy to optimize. This can speed up the optimization process related to existing methods for obtaining a global solution point.
2. The proposed optimization models are easy to implement in a symbolic format in YALMIP/MATLAB, and BBA makes it easy to optimize those models.
3. The time spent on the entire optimization process illustrates that the procedure has low time complexity in achieving optimality.
4. The proposed models are tested on 14-, 30-, 57, and a medium-size model of the IEEE-118 bus system. This fact enacts the tendency to extend the OPP to larger power systems.

Algorithms are symbolically implemented in MATLAB, with the model being optimized while the optimization is executed by YALMIP. The primal and secondary optimization models are solved through a BBA solver built-in in YALMIP calling external routines [77,78]. The primal and secondary problems are characterized by their solution robustness given by the log file of the BBA embedded in the MATLAB function, such as the MATLAB® function intlinprog [70], the SCIP optimizer [72], and the Gurobi function [74].

IEEE 14-, 30-, 57-, and 118-bus network systems are utilized for the implementation of the proposed optimization models performed in the MATLAB environment [70–74,77,80]. We utilize a symbolic language using the YALMIP library to build the proposed optimization models [69,76–78].

As a result, it is based on common sense to validate the proposed minimization and maximization models on the classical IEEE power systems to show their implementation based on convergence and stopping criteria [44].

The resulting solution covers a solution within a 0.00% optimality criterion, staying within specific tolerances such as absolute and relative gaps [9–11,70,72].

The solution of the maximization problem shows that the maximum network observability is preserved [22,34,35,39,41]. The resulting solutions have specific maximum indices as proposed in [22], and a comparative study is conducted with optimality metrics presented in [35,46]. BPSO optimizes a two-product objective function, whereas GPSA minimizes a cost function with one criterion [35,46].

Also, GSA is utilized in [34] to optimize the OPP problem. A two-objective function is well optimized in order to locate those optimal solutions that satisfy the observation of the entire power network while preserving maximum network observability [22,33–35,39,41,46].

Although a 0/1 semi-definite programming formulation is proposed in [28], its capability to achieve optimal results in terms of maximum network observation of the power grid state must be reconsidered [22,34,35,39,41,46].

Although the SCIP is utilized with CUTSDP as an outer approximation optimizer function to solve the SDP relaxations, the optimal solution is not feasible in terms of maximum network observation [34,35,39,41]. All optimization models are symbolically implemented using the YALMIP program [69] with powerful advanced optimization solvers to obtain an optimal solution within a 0.00% criterion [74].

To prove such an optimality criterion being achieved by the algorithm's run, the log file is given in the Appendices C and D following the main body of this work [70–72].

As a result, all instances were solved using an optimality criterion equal to 0.00%. That means that no higher-quality solution can be found. Hence, reliable power system monitoring is now attainable by synchronized measurements [77].

Considering that our approach is to compare the performance of several global ILP solvers to the YALMIP problem-based optimal PMU placement implementation, we illustrate all the solution outcomes tabulated in relative tables in the Appendices B–E.

7.1. Results of Complete Observability of the Electricity Power Transmission Grid

The present study aims at minimizing an objective function to find a global solution point, meaning that this solution satisfies the by-default tuning or tolerance parameters. As a result, the outcome of each state-of-the-art solver gives an optimal solution within a zero-gap absolute gap as well as a zero price percentage relative gap [10–12].

We used the MATLAB R2018a [70,71] to optimize the mathematical models in collaboration with the SCIP optimization suite 8.0 [72,73], Gurobi 10.0.1 [74] and MOSEK 10.1 [79]. All optimization models are symbolically programmed with YALMIP 20210331 [69,76,78] and giving an optimal outcome.

Simulation results for the minimization problem applied to benchmark IEEE power systems [80] have been presented to illustrate that an optimal solution is achieved with feasibility and stopping criteria to be satisfied [44]. The PMU numbers needed for full observation of the power grid estimation and their positioning sites are shown in the tables in the Appendices B–E.

7.2. Results of Maximum Observation of the Electricity Power Transmission Grid

The main target is to make the state estimation process fully functional. PMU monitoring devices must be located strategically at power network nodes with appropriate placement sites and thereby succeed at maximizing the coverage of the obtainable measurement redundancy [22,33–35,39,41,46].

A unified effort has been spent to highlight a compact solution obtained by a primary-secondary optimization process. First, we minimize the total investment cost of PMUs to minimize the entire set of PMUs to be desired [33,45,46].

An optimal solution is attained through the solution of ILP, SDP, and polynomial problems with Boolean consideration [9–11]. Then, a maximization problem follows the solution of the minimization problem with an extra constraint function reflecting the optimal solution satisfying the criterion of full observation of the power grid state.

Solving such a kind of optimization problem, we derive solutions with maximum network observability. Each maximization model is solved by comparing several global ILP solver outcomes. As we can see, the outcomes are different, and they depend on a different search order in building the enumeration tree while an optimal solution is achieved [9–11].

All maximization problems are exactly solved with the suboptimality criteria fully met [9–11]. Simulation results for maximization problems applied to standard IEEE power systems have been reported to illustrate that an optimal solution is achieved with feasibility and stopping criteria satisfied [44]. The placement results with the maximum SORI are shown in relative tables in the Appendices B–E.

We return those solutions with maximum SORI. The selected strategic sites maximize measurement redundancy. The maximum observability indicators for each power network are illustrated in Table A1. The same indices are found in [22,34,35,39,41,46].

With the maximum values of the SORI for the suggested maximization problem, it is clearly deduced that the proposed algorithms would count the maximum measurement redundancy. This truth validates our extended work in this research domain related to the maximum reliability of measurements in the state estimator process [1,2,59–63].

Appendices B–E illustrates the simulation results derived for both case studies namely full network observability and maximum network observability as well [20–22]. For each IEEE system, the best objective function value is illustrated with minimum and maximum SORI as defined in [22].

8. Discussion

The PDS, known as the OPP problem [3–5], is one of the greatest mathematical formulations studied, belonging to the combinatorial optimization problems [9–18].

Our proposal gives a structured procedure for solving the discrete optimization model in mathematical domains such as ILP [20–23], SPD [26–28], and Boolean polynomial models [9,10,44].

For that purpose, minimization models such as ILP [20–23] and SDP [26–28] are reconsidered to obtain a necessary and sufficient condition for optimality, and a quadratic Boolean polynomial is also proposed [9,10,42,44].

Then, a maximization model is implemented, taking into account the global solution achieved by the minimization problem, to deliver those solutions with SORI [22,46].

Initially, a minimization model is optimized to find the optimal number of PMUs, and then, following that number, a maximization model is constructed to give this solution, which satisfies the maximum network observation of the power state grid [22,46].

The former model is a binary integer linear programming model that can be solved by any powerful advanced MILP solver. Then, a CBQP model is utilized that is fully equivalent to the former integer linear programming model with binary values [10].

Both algorithmic models are solved by the YALMIP BBA built-in BMIBNB routine [77]. We utilize this routine to solve the CQBO model and then a linear CBO to achieve the placement of PMU sets with the highest SORI [22,34,35,39,41,46].

BMIBNB is based on external MILP optimizer functions to solve the lower bounding relaxation programming problems and a nonlinear function to estimate the upper bound.

The lower solver is used to solve the relaxed problems, the local solver to count the upper bound, and an LP solver to calculate the bound strengthening [77].

The upper bound is produced by invoking the upper solver; the relaxed problems are implemented by an LP solver; and a lower bound is generated with a lower solver constantly compared with the upper bound. The solution is found to be inside a zero-valued absolute and relative gap, and that means no higher-quality solution can exist.

As a result, it is characterized as a globally optimal solution. Also, a Boolean SDP is reconsidered by using a CUTSDP solver and powerful optimizer solvers such as MATLAB® function intlinprog [70], SCIP optimizer function [72,73], Gurobi function [74], and MOSEK [79] to obtain an optimal solution.

The algorithm returns a global optimum objective function value and its corresponding binary-valued decision vector to the decision-making problem.

BBA terminates within specific tolerances and gives the global solution for 0/1 ILP [20–23], 0/1 SDP [26–28], and 0/1 constraint polynomial problems [9,10].

Despite the notation that the ILP model is trapped in the same solution and cannot handle the task of measurement redundancy [24,36], we utilize the GUROBI to implement it and collect all the optimal solutions [74].

We illustrate that the BBA gives all the solutions using the solution pool option of GUROBI for both case observability scenarios, as illustrated in Figure 1 [74].

As a result, our innovation, besides the fact that the solution is a global outcome discovered inside a zero-gap tolerance, is that we deliver all the optimal solutions through binary integer linear programming [9–11]. Also, we bring to your attention that the same number of PMUs has been found in [25] for the IEEE-30 bus system [80] to further validate our approach.

The total number of PMUs and the number of PMU placements as well are shown in Figure 1 for the complete observability scenario. The total number of PMUs is produced by optimizing the binary integer program using the GUROBI optimizer [74].

GUROBI solves exactly the optimization problem within a zero-gap tolerance, giving all the optimal solutions as shown in Figure 1 and the relative tables in the Appendix E.

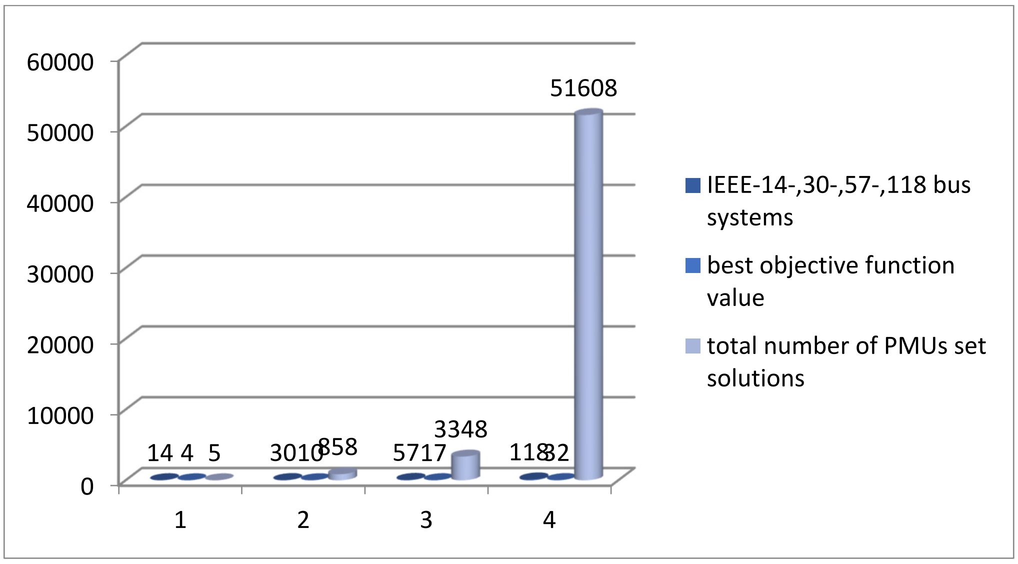

Figure 1. Total PMU set solutions for the complete observability index.

In addition, we declare a maximization problem by which the ILP can handle with efficiency the measurement redundancy task. Figure 2 illustrates the minimum and maximum SORI observability indicators produced by optimizing the binary integer, Boolean semi-definite, and constraint polynomial models [22,46].

This chart shows that our methodology is superior to the work published in [28,50]. The values of the maximum observability indicator as shown in Figure 2 are in full agreement with those found in [9–11,34,35,41,46].

The minimum values of SORI are produced by minimizing the maximization problem. As a result, our study illustrates the best outcomes related to this observability index, as proposed in [22]. Figure 3 illustrates the total number of PMU set solutions with the maximum observability index [22,35,39,46].

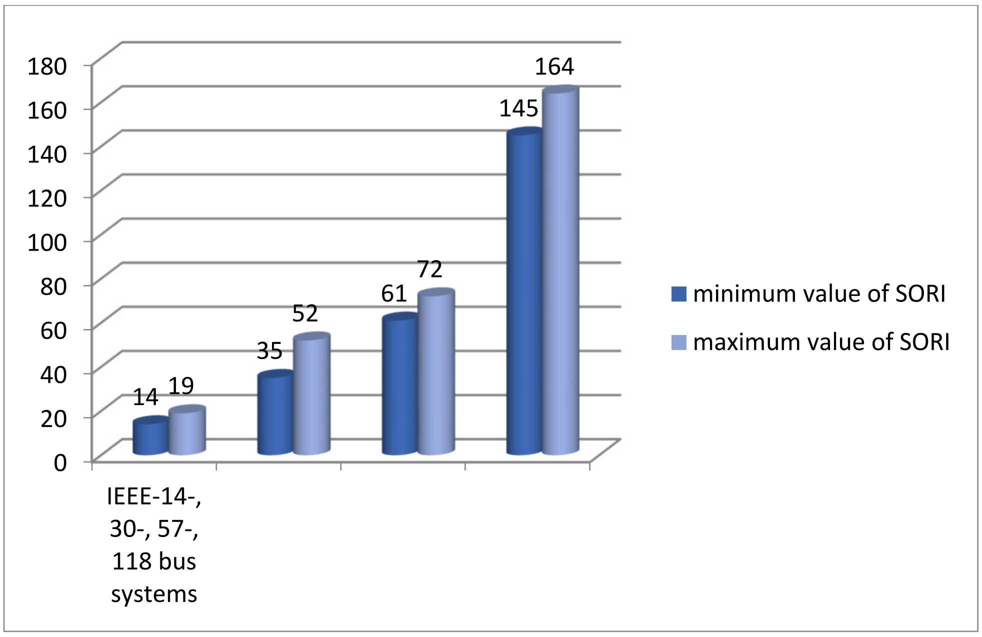

Figure 2. SORI of binary integer, Boolean semi-definite, and polynomial models.

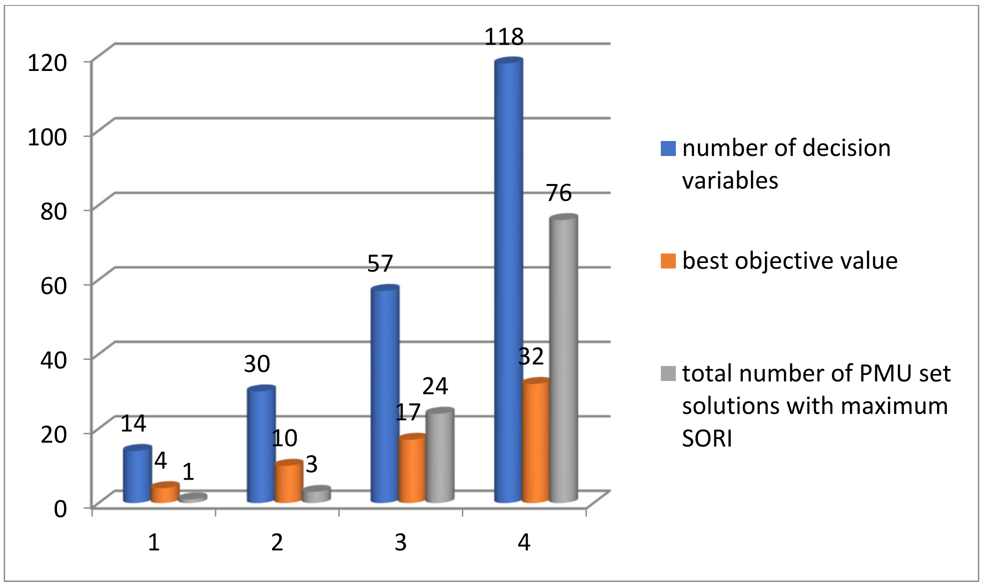

Figure 3. Total PMU set solutions having the maximum observability index.

Placing an exact number of PMUs helps the system operator avoid abnormal situations and guarantees a constant supply of power and electricity to the utilities and customer satisfaction (Figure 1). The total number of PMUs is derived using the GUROBI optimizer engine, which adjusts the solution pool option properly [9,11,74].

Also, the best number of PMUs can be posed at specific power grid nodes to satisfy the measurement redundancy, as shown in Figure 3. As a consequence of this, a power grid node will be observed the maximum possible number of times, either directly or indirectly, by the laws of Kirchhoff and Ohm [19–23,32,45,46].

9. Conclusions

In this work, two goals have been studied to satisfy a need to observe the power grid state with the declaration of one objective function representing the total cost of PMU installation. The aim is to decrease the PMU numbers to satisfy the observation of the electricity power transmission grid.

The second goal is to maximize the power system observation of the power grid through the declaration of a maximization problem. The minimization problem is considered the primal optimization, which gives a global solution point.

This solution is used as an extra constraint in a maximization problem to derive those solutions that are best and satisfy the maximum reliability of the electricity power transmission grid at the same time.

For the minimization problem, optimization of a linear objective function was conducted under ILP, polynomial, and LMI constraints. The primary outcome is the discovery of a binary-valued optimum point. We implement optimization models using a YALMIP library fully compatible with MATLAB. Our main solution algorithm is the branch-and-bound algorithm, which returns a globally high-quality solution for each optimization model. Optimizing the tuning parameters of the BBA, solutions are derived based on the accuracy of an optimal criterion equal to 0.00%. Illustrating its robustness, a solution was achieved by meeting a 0.00% optimality criterion.

In the second step of the process, the consideration shifts towards maximization problems, adhering to the logic of primal and secondary solutions. Leveraging the high-quality solution derived from the minimization problem, the aim is to identify a solution that satisfies the requirement for comprehensive monitoring across a wide area from the perspective of maximizing network observability.

The implementation of the proposed methods, either in minimization or maximization problems unambiguously programmed in MATLAB and easy to solve, proved the iterations of the BBA to achieve the desired constraint minimum point under sufficient global optimality conditions.

As a result, both optimization schemes are well optimized at finding constraint minimum points satisfying indisputable optimality conditions. BBA succeeds in building a binary tree, explores a suitable number of nodes, solves linear problem relaxations, and finally terminates the entire optimization with the exact solution.

We find out that the upper bound is the optimal objective value of the minimization problem at the final root node, whereas the lower bound is equal to the upper bound, closing the absolute gap. As a result, the BBA terminates with a stopping criterion to be met. For the maximization problem, the lower bound is considered the best solution.

All models are solved by building a binary tree as small as possible, and the solution is found inside a zero-valued absolute and relative gap.

These stopping criteria are necessary and sufficient conditions to show if the solution is a global solution point or a suboptimal outcome. We accomplish our target, and the optimization problems are solved globally.

As a result, we fill the knowledge gap about OPP problem solving and whether it derives optimal or suboptimal solutions. These algorithms perform with high accuracy and optimize small and medium-sized optimization problems without losing the chance of determining the global optimum as the size of the dimensions increases.

To enhance our proposal, log files are given in the Appendices C–E. As the log files illustrate, the binary search tree is small in size due to the simplicity of the symbolic reformulation of the proposed optimization models.

Our proposal gives a trade-off optimization run in terms of the implantation of the best optimal solution in the first stage and the second solution satisfying maximum reliability. In summary, an actual-world problem is stated in a decision-making way as a solution without facing any uncertainty about optimality.

10. Further Research Based on the Topics Studied in This Study

Minimizing the number of PMUs reduces the entire investment cost of PMU installation needed to fully monitor an electricity power transmission system, which is the principal concern for strategically posing PMUs around the power transmission grid.

The task studies in this work contain SynchroPhasors for the performance of the power grid; the best reliability of the system observability redundancy index (SORI) for the best arrangement of PMUs around the power grid; and complete and maximum observation of the power grid.

At the moment, PMUs are integrated with the SCADA system to observe the power grid state. In the recent past, artificial intelligence (AI) and machine learning (ML) methods were used for the detection of abnormal situations occurring in the electricity transmission power grids, such as fault events.

These algorithms are able to detect, recognize, and locate the fault occurring on a transmission line. The strategic implications of our proposed method cover a wide range of methodological and applied contributions. The planning consequences of our proposed algorithmic approach are summarized in the following items:

PMUs are technology-advanced devices designed to observe power flows regarding power generation, demand, turn-on features, and advanced system management.

a. The future grid will have the capability to incorporate a large number of PMUs to help a lot with utilities and customers' requirements for constant power supply at the lowest possible cost.
b. We propose those PMU arrangements that improve power grid reliability through the maximum observation of the power grid state estimator tool installed in the system operator.
c. It produces a complete working OPP solution with accuracy and improved ability to accomplish the aim for the exact number of PMUs under guarantee as well as those solutions satisfying maximum reliability.
d. A future direction is to extend our symbolic programming models and computational way of their solutions under any contingencies in SGs.
e. Future research could investigate the combination of PMUs with AI and ML algorithms to enable real-time detection and response to cyberattacks.
f. It also works as a first test for a novel algorithmic scheme to use a global optimizer routine for solving combinatorial problems with the symbolical programming language and finding a global solution point.

Author Contributions: Conceptualization, N.P.T.; methodology, N.P.T.; software, N.P.T.; validation, N.P.T.; investigation, N.P.T.; resources, N.P.T.; data curation, N.P.T.; writing—original draft preparation, R.B.; validation, editing—review, A.P.M.; supervision. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The authors confirm that the data supporting the findings of this study are cited within the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SCADA	Supervisory Control and Data Acquisition
PMU	Phasor Measuring Unit
PDC	Phasor Data Concentrator
SG	Smart Grid
WAMS	Wide Area Monitoring Systems
PDC	Power Dominating Set
OPP	Optimal PMU Placement
SORI	System Observability Redundancy Index
BOI	Bus Observability Index
MILP	Mixed-Integer-Linear Program
QCBO	Quadratic Constrained Binary Optimization
LCBO	Linear Constrained Binary Optimization
SCIP	Solve Constraint Integer Program
BBA	Branch-and-Bound Algorithm
s-BBA	Spatial Branch-and-Bound Algorithm
TolGapAbs	Tolerance Gap Absolute
MIPGapAb	Mixed-Integer Program Gap Absolute

Appendix A

For further confirmation of our results, we present the log file output in the Appendices C–E. Initially, we present the optimization of the ILP model, either in a minimization or maximization model. This optimality condition is a zero-valued function gap [11].

The upper bound is the desired solution for the minimization problem, whereas the lower bound is the proper output for the maximization problem [9,11,74].

As a result, the return solution is within this default tolerance and can be characterized either as an optimal solution by MATLAB® function intlinprog [72] or as a global solution by the log file of the SCIP optimizer [72] or MOSEK optimization library [79].

Also, binary quadratic polynomial primal and secondary linear polynomial problems are solved by the BBA in a similar computational manner. The primal and secondary problems are well-established and programmed in the YALMIP program, dealing with the issue of finding a solution with an absolute zero gap tolerance [69]. Hence, a global solution point has been achieved for all optimization models [9–11,70].

The proposed polynomial models are symbolically programmed in YALMIP, which performs as a MATLAB library, and a built-in BBA solver implements the whole optimization [69]. This integer solver calls external solvers such as MATLAB® function intlinprog [70] and MATLAB® function fmincon [71] to obtain an optimal solution [9–11].

To achieve this, bound tightening is turned on in all instances. A global solution point means that it satisfies the MIPGapAb that is adjusted to be equal to zero [70].

Yet again, the solution is computationally attainable, and one is concerned with avoiding suboptimality uncertainties. As a result, a solution that is obtainable with a 0.00% optimality criterion is considered to be a global outcome [9–11,70].

Also, CUTSDP [78], a built-in solver in YALMIP, solves the 0/1 SDP program either in minimization or maximization using a cutting-plane strategy using advanced technology solvers like MATLAB® function intlinprog [72] or as a global solution by the log file of SCIP optimizer routine [72] or MOSEK optimization library [79].

Appendix B

Appendix B illustrates the simulation results derived for both case studies namely full network observability and maximum system observability index (SORI) as well [22].

Table A1. Best objective function value for standard IEEE system with minimum and maximum SORI.

IEEE-Power Systems	PMUs Number	Minimum SORI	Maximum SORI
IEEE-14 bus	4	14	19
IEEE-30 bus	10	35	52
IEEE-57 bus	17	61	72
IEEE-118 bus	32	145	164

Table A2. Optimal PMU positioning sites derived by solving the pure minimization MILP model with YALMIP software.

Mixed-Integer-Linear-ProgramMixed-Integer-Linear-Program Implementation in YALMIP Software.			
IEEE-Systems	**GUROBI**	**SCIP**	**Intlinprog**
	Optimal PMU Locations		
14-bus	2, 7, 10, 13	2, 7, 11, 13	2, 8, 10, 13
30-bus	1, 5, 6, 9, 10, 12, 15, 19, 25, 27	2, 4, 6, 9, 10, 12, 15, 19, 25, 30	1, 5, 8, 10, 11, 12, 19, 23, 26, 29
57-bus	2, 6, 12, 19, 22, 25, 27, 29, 32, 36, 39, 41, 45, 46, 49, 51, 54	2, 6, 12, 14, 19, 22, 25, 27, 32, 36, 39, 41, 44, 47, 50, 52, 55	1, 4, 9, 20, 23, 27, 29, 30, 32, 36, 38, 41, 45, 46, 50, 54, 57
118-bus	1, 5, 9, 12, 15, 17, 21, 23, 28, 30, 36, 40, 44, 46, 50, 52, 56, 62, 63, 68, 71, 75, 77, 80, 85, 86, 91, 94, 102, 105, 110, 114	3, 5, 9, 11, 12, 17, 21, 25, 28, 34, 37, 40, 45, 49, 52, 56, 62, 63, 68, 70, 71, 75, 77, 80, 85 86, 90, 94, 101, 105, 110, 114	2, 5, 10, 12, 15, 17, 21, 25, 29, 34, 37, 41, 45, 49, 53, 56, 62, 64, 72, 73, 75, 77, 80, 85, 87, 91, 94, 101, 105, 110, 114, 116

Table A3. Optimal PMU positioning sites derived by solving the pure maximization MILP model with YALMIP software.

Mixed-Integer-Linear-Program Implementation in YALMIP Software.			
IEEE-Systems	**GUROBI**	**SCIP**	**Intlinprog**
	Optimal PMU Locations		
14-bus	2, 6, 7, 9	2, 6, 7, 9	2, 6, 7, 9
30-bus	2, 4, 6, 9, 10, 12, 15, 20, 25, 27	2, 4, 6, 9, 10, 12, 15, 18, 25, 27	2, 4, 6, 9, 10, 12, 15, 19, 25, 27
57-bus	1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 41, 46, 51, 53, 57	1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 39, 41, 46, 50, 53	1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 41, 46, 50, 53, 57
118-bus	3, 5, 9, 12, 15, 17, 21, 25, 28, 34, 37, 40, 45, 49, 53, 56, 62, 64, 68, 70, 71, 78, 85, 86, 89, 92, 96, 100, 105, 110, 114, 118	3, 5, 9, 12, 15, 17, 21, 25, 28, 34, 37, 40, 45, 49, 52, 56, 62, 64, 68, 70, 71, 76, 78, 85, 86, 89, 92, 96, 100, 105, 110, 114	3, 5, 9, 12, 15, 17, 21, 25, 29, 34, 37, 40, 45, 49, 53, 56, 62, 64, 68, 70, 71, 75, 77, 80, 85, 86, 91, 94, 101, 105, 110, 114

Table A4. Optimal PMU positioning sites derived by solving the QCBO model with YALMIP software.

Mixed-Integer-Linear-Program Implementation in YALMIP Software.			
IEEE-Systems	**GUROBI**	**SCIP**	**Intlinprog**
	Optimal PMU Locations		
14-bus	2, 8, 10, 13	2, 6, 8, 9	2, 7, 11, 13
30-bus	1, 5, 8, 10, 11, 12, 19, 23, 26, 29	1, 2, 6, 10, 11, 12, 18, 23, 26, 27	3, 5, 8, 10, 11, 12, 15, 20, 25, 30
57-bus	1, 4, 9, 20, 24, 28, 29, 30, 32, 36, 38, 39, 41, 44, 46, 50, 53	2, 6, 12, 19, 22, 25, 27, 32, 36, 39, 41, 45, 46, 47, 50, 52, 55	3, 6, 12, 15, 19, 22, 25, 26, 29, 32, 36, 38, 41, 46, 50, 54, 57
118-bus	3, 5, 10, 12, 15, 17, 21, 23, 28, 30, 36, 40, 43, 45, 49, 52, 56, 62, 64, 71, 75, 77, 80, 85, 87, 91, 94, 101, 105, 110, 114, 116	2, 5, 10, 12, 15, 17, 20, 23, 29, 30, 36, 40, 43, 46, 51, 54, 57, 62, 64, 71, 75, 77, 80, 85, 87 91, 94, 102, 105, 110, 115, 116	3, 5, 10, 12, 15, 17, 21, 25, 28, 34, 37, 42, 45, 49, 52, 56, 62, 64, 71, 72, 75, 77, 80, 85, 87, 90, 94, 102, 105, 110, 114, 116

Table A5. Optimal PMU positioning sites derived by solving the LCBO model with YALMIP software.

Mixed-Integer-Linear-Program Implementation in YALMIP Software.			
IEEE- Systems	**GUROBI**	**SCIP**	**Intlinprog**
	Optimal PMU Locations		
14-bus	2, 6, 7, 9	2, 6, 7, 9	2, 6, 7, 9
30-bus	2, 4, 6, 9, 10, 12, 15, 20, 25, 27	2, 4, 6, 9, 10, 12, 15, 18, 25, 27	2, 4, 6, 9, 10, 12, 15, 19, 25, 27
57-bus	1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 39, 41, 47, 51, 53	1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 41 46, 50, 53, 57	1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 41, 46, 51, 53, 57
118-bus	3, 5, 9, 12, 15, 17, 21, 25, 29, 34, 37, 40, 45, 49, 52, 56, 62, 64, 68, 70, 71, 78, 85, 86, 89, 92, 96, 100, 105, 110, 114, 118	3, 5, 9, 12, 15, 17, 21, 25, 29, 34, 37, 40, 45, 49, 52, 56, 62, 64, 68, 70, 71, 75, 77, 80, 85 86, 90, 94, 102, 105, 110, 114	3, 5, 9, 12, 15, 17, 21, 25, 28, 34, 37, 40, 45, 49, 52, 56, 62, 64, 68, 70, 71, 78, 85, 86, 89, 92, 96, 100, 105, 110, 114, 118

Table A6. Optimal PMU positioning sites derived by solving the SDP minimization model with YALMIP software.

IEEE-Systems	**MOSEK**	**SCIP**
14-bus	2, 6, 7, 9	2, 7, 10, 13
30-bus	1, 2, 6, 9, 10, 12, 15, 18, 25, 27	1, 2, 6, 9, 10, 12, 15, 19, 25 30
57-bus	1, 6, 13, 15, 19, 22, 25, 26, 29, 32, 36, 38, 41, 46, 51, 54, 57	2, 6, 12, 14, 19, 22, 25, 27, 32 36, 39 41, 44, 47, 50, 52, 55
118-bus	1, 6, 9, 11, 12, 17, 21, 25, 29, 34, 37, 41, 45, 49, 53, 56 62, 63, 68, 71, 72, 75, 77, 80 85, 86, 90, 94, 102, 105 110, 114	3, 5, 9, 11, 12, 17, 21, 25, 28, 34, 37 40, 45, 49, 52, 56, 62, 63, 68, 70, 71 75, 77, 80, 85, 86 90, 94, 101, 105, 110, 114

Table A7. Optimal PMU positioning sites derived by solving the SDP model maximization with YALMIP software.

IEEE-Systems	**MOSEK**	**SCIP**
14-bus	2, 6, 7, 9	2, 6, 7, 9
30-bus	2, 4, 6, 9, 10, 12, 15, 19, 25, 27	2, 4, 6, 9, 10, 12, 15, 20, 25, 27
57-bus	1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 41, 46 50, 53, 57	1, 4, 6, 9, 15, 24, 25, 28, 32, 36 38, 41 46, 51, 53, 57
118-bus	3, 5, 9, 12, 15, 17, 20, 23, 28, 30, 34, 37, 40, 45, 49, 52, 56, 62, 64, 68, 71, 75, 77, 80 85, 86, 90, 94, 102, 105, 110, 114	3, 5, 9, 12, 15, 17, 21, 25, 29, 34 37, 40 45, 49, 53, 56, 62, 64, 68, 70, 71, 75, 77, 80, 85, 86, 90, 94 101, 105 110, 114

Table A8. Optimal PMU positioning sites derived by solving the pure minimization SDP model with YALMIP software using dual-simplex for linear relaxations.

IEEE-Systems	**Intlinprog**	**Gurobi**
14-bus	2, 6, 7, 9	2, 6, 7, 9
30-bus	1, 5, 8, 10, 11, 12, 19, 23, 26, 29	1, 5, 6, 9, 10, 12, 15, 19, 25, 27
57-bus	1, 4, 9, 20, 23, 27, 29, 30, 32, 36, 38, 41, 45, 46, 50, 54, 57	2, 6, 12, 19, 22, 25, 27, 29, 32, 36, 39, 41, 45, 46, 49, 51, 54
118-bus	2, 5, 10, 12, 15, 17, 21, 25, 29, 34, 37, 41, 45, 49, 53, 56, 62, 64, 72, 73, 75, 77, 80, 85, 87, 91, 94, 101, 105, 110, 114, 116	1, 5, 9, 12, 15, 17, 21, 23, 28, 30, 36, 40, 44, 46, 50, 52, 56, 62, 63, 68, 71, 75, 77, 80, 85, 86, 91, 94, 102, 105, 110, 114

Table A9. Optimal PMU positioning sites derived by solving the pure minimization SDP model with YALMIP software using primal-simplex for linear relaxations.

IEEE-Systems	Intlinprog	Gurobi
14-bus	2, 6, 7, 9	2, 6, 7, 9
30-bus	1, 7, 8, 10, 11, 12, 19, 23, 25, 29	1, 5, 6, 9, 10, 12, 15, 19, 25, 27
57-bus	1, 4, 7, 9, 20, 22, 25, 27, 32, 36, 39, 41, 44, 46 49, 50, 53	2, 6, 12, 19, 22, 25, 27, 29, 32, 36, 39, 41, 45, 46, 49, 51, 54
118-bus	1, 6, 9, 11, 12, 17, 21, 25, 28, 34, 37, 4,1 45, 49 52, 56, 62, 63, 68, 70, 71, 78, 85, 86, 91, 92, 96 100, 105, 110, 114, 118	1, 5, 9, 12, 15, 17, 21, 23, 25, 28, 34, 37, 40, 45, 49, 52, 56, 62, 63, 68, 71, 75, 77, 80, 85, 86, 91, 94, 102, 105, 110, 114

Table A10. Optimal PMU positioning sites derived by solving the pure maximization SDP model with YALMIP software using dual-simplex for linear relaxations.

IEEE-Systems	Intlinprog	Gurobi
14-bus	2, 6, 7, 9	2, 6, 7, 9
30-bus	2, 4, 6, 9, 10, 12, 15, 19, 25, 27	2, 4, 6, 9, 10, 12, 15, 19, 25, 27
57-bus	1, 4, 6, 9, 15, 20, 24, 28, 31, 32 36, 38, 41, 46, 50, 53, 57	1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38 41, 46, 51, 53, 57
118-bus	3, 5, 9 12, 15, 17, 21, 25, 29, 34 37, 40, 45, 49, 53, 56, 62, 64, 68 70, 71, 75, 77, 80, 85, 86, 91, 94 101, 105, 110, 114	3, 5, 9, 12, 15, 17, 21, 25, 29, 34, 37, 40, 45, 49, 53, 56, 62, 64, 68, 70, 71, 75, 77, 80, 85, 86, 90, 94, 101, 105, 110, 114

Table A11. Optimal PMU positioning sites derived by solving the pure maximization SDP. model with YALMIP software using primal-simplex for linear relaxations.

IEEE-Systems	Intlinprog	Gurobi
14-bus	2, 6, 7, 9	2, 6, 7, 9
30-bus	2, 4, 6, 9, 10, 12, 15, 20, 25, 27	2, 4, 6, 9, 10, 12, 15, 20, 25, 27
57-bus	1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 41, 47 50, 53, 57	1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 41, 47, 51, 53, 57
118-bus	3, 5, 9, 12, 15, 17, 20, 23, 29, 30, 34, 37, 40, 45, 49, 52, 56, 62, 64, 68, 71, 75, 77, 80, 85, 86, 91, 94, 102, 105, 110, 115	3, 5, 9, 12,15, 17, 20, 23, 28, 30, 34, 37, 40, 45, 49, 53, 56, 62, 64, 68, 71, 75, 77, 80, 85, 86, 90, 94, 102, 105, 110, 115

Appendix C

In Appendix C, we report the BBA solver's progress by illustrating the optimization function log files related to the minimization problems. Initially, we run the 0/1 ILP model, then transform it into a 0/1 SDP model, and finally, we present the QCBO for a representative small-sized power system [80].

The search log delivers extensive knowledge of searching during the model optimization. We present three samples of the search log for the minimization model.

Each log points to the type of the optimization model, in addition to the number of design variables and the total number of constraints in each model [11].

All log files present convergence at the constraint minimum point. They illustrated that the suggested BB algorithm is able to detect a global optimum with an acceptable convergence performance of the iteration process [9–11,70–79].

We present the log files as they are given by optimizing the models through the built-in YALMIP optimizer solvers [69,76–78]. The special characters show the optimization during the iterative process produced by the routines towards optimality [69,76–78].

The absolute and relative gap is an adequate optimality condition that leads to the conclusion that a globally optimal solution has been achieved for each minimization model [9–11,44].

In the minimization problems studied in this work, an upper bound on the optimum solution point is attained by any feasible solution. This upper bound is the tightest limit, and it is the best incumbent solution found so far [70–79].

In all cases, the absolute as well as the relative gap are computed to be zero. As a result, the determination of finding a global solution is proved by the log files of the routine. The upper bound is the global solution for the minimization problem [9–11].

I. Mixed-Integer Linear Programming Model

The optimization run is performed, taking into account a number of algorithmic results measured during the iterative process. The whole optimization is met when specific feasibility, tolerance, and termination criteria are fully satisfied [44].

The YALMIP BBA solver interfaces with an external MILP solver, builds the BBA tree, searches the feasible region, prunes infeasibilities, utilizes heuristic computations, and returns a global solution point. Heuristic calculations are used to improve the upper bound. ILP solvers try to deliver a branch-and-bound tree as small as possible without producing many branches [9–11]. As a result, a branching regulation with strength is executed, which results in a solution that satisfies the optimality conditions [64–67].

The IEEE-30 bus system is used as a case study to show the applicability of YALMIP BBA to the 0/1 MILP model [80]. PMU placement is shown in Tables A12 and A13, whereas the absolute and relative gaps are shown in Table A14. An optimal solution is to return to the final given root node of the BBA tree within a 0.00% criterion [76–78].

Table A12. Mixed-Integer Programming Solver Log File.

```
+ Solver chosen: BMIBNB
+ Processing objective function
+ Processing constraints
+ Branch and bound started
* Starting YALMIP global branch and bound.
* Upper solver:   GUROBI
* Lower solver:   GUROBI
* LP solver:      GUROBI
* -Extracting bounds from model
* -Performing root-node bound propagation
* -Calling upper solver + Calling GUROBI
(found a solution!)
* -Branch-variables: 0
* -More root-node bound-propagation
* -Performing LP-based bound-propagation
* -And some more root-node bound-propagation
* Starting the B and B process
Node   Upper   Gap(%)   Lower   Open   Time
+ Calling GUROBI
1:   1.00000E+01   0.00   1.00000E+01   0   0s   Poor lower bound
* Finished. Cost: 10 (lower bound: 10, relative gap 0%)
* Termination with all nodes pruned
* Timing: 17% spent in upper solver (1 problems solved)
*   1% spent in lower solver (1 problems solved)
*   1% spent in LP-based domain reduction (0 problems solved)
*   1% spent in upper heuristics (0 candidates tried)
```

Table A13. The PMU placement sites with the best objective value and a SORI.

Optimal PMU Placement Sites
1, 5, 6, 9, 10, 12, 15, 19, 25, 27
Optimal number of PMUs: 10 SORI: 48 best function value: 10

Gurobi computes both the upper and lower bounds on the objective function value to find the optimal solution [70]. Gurobi utilizes the TolGapAbs, which is the difference between the aforementioned bounds and is used as a stopping criterion [69–74].

The algorithm ends up with a certificate giving a ε-suboptimality criterion [9–11]. Also, the relative gap is calculated to be zero where the termination ends with all nodes pruned [69]. Thus, suboptimal solutions are avoided [70].

The optimizer function considers, without stopping the upper and lower bounds on the objective function, the difference between those bounds. An absolute gap is computed and confirms the entire optimization towards optimality [70].

A solution is considered to be a global outcome when the objective function value is found to be within a prescribed gap tolerance criterion [69–74].

As the log file shows, the absolute gap as well as the relative gap are equal to 0.00%, which means it is not possible to find a high-quality solution. As a result, a globally optimal solution is achieved under warranty [69–74].

As the log file shows in the Appendices C–E the upper bound leads the optimization run, with a lower bound equal in quantity, so that the difference between those bounds can be estimated as a zero-valued price [76]. As a result, the outcome is a global solution [9–11,70,72–74,77–79].

The plot diagram is illustrated in Figure A1. As a result, the absolute as well as the relative gap tolerance are discovered to be equal to zero, which guarantees that global optimality is delivered at the final root node of the BBA tree [76,77].

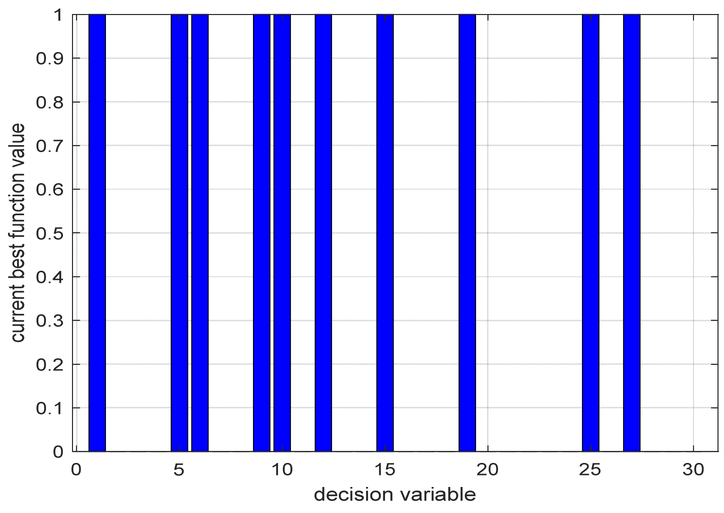

Figure A1. Optimal PMU positioning sites under complete observability.

II. Quadratic Constrained Binary Optimization Model

We optimize a quadratic objective function subject to a polynomial constraint function in a binary symbolic format. Our hope is to show a new impact on the field of non-convex optimization. A standard global BBA optimizer embedded in YALMIP is utilized, and an upper bound is needed to be calculated for the minimization problem.

It must be noted that the proposed BB algorithm is able to ensure a global solution point despite the fact that the nonconvex subproblems are solved to local optimality using nonlinear algorithms. Our innovation is the reformulation in symbolic format with binary decision variables, solved globally by a BBA scheme [77].

We optimize the model in YALMIP to return the best objective upper bound. We utilize a local nonlinear optimization function to compute the upper bounds by utilizing the command '*bmibnb.uppersolver*' [77].

In addition to that strategy, an integer linear function is used to estimate the lower bounds of the objective function value by invoking the command '*bmibnb.lowersolver*' [77].

Also, the relaxed linear programming problems and bound tightening tasks are solved by calling an LP solver with the command '*bmibnb.lpsolver*' [71]. On the other side, a MILP solver is invoked by the YALMIP BBA to solve the linear programming relaxations, tighten the boundaries, and find the lower bound. Calculation time is spent on solving the linear programs based on bound propagation. BMIBNB executes this run for such optimization problems [77].

By solving the minimization model, the upper bound is considered the optimum point, while the lower bound closes the absolute gap. The message status is as follows:

The algorithm ends when the difference between the upper and lower bounds is decreased below the $\varepsilon - tolerance$ [77].

The declaration of the stopping criterion leads to a finite termination. Due to the nature of the minimization models, the solution is returned within a zero-value gap tolerance. If the termination criterion $u_b - l_b \leq \varepsilon$ is fulfilled, then the algorithm terminates the entire optimization run and an ε-globally optimal value has been achieved [9–11].

The optimizer function constantly calculates the upper and lower bounds on the cost function value and the difference between those bounds [77]. The algorithm ends up with an optimality certificate. The IEEE-30 bus system is used as a case study to show the applicability of YALMIP BBA to the 0/1 quadratic constraint polynomial model [80]. Table A14 illustrates the repetitive strategy to produce a global solution point within the predefined tolerances [9–11,70–79].

An absolute gap is estimated and confirms the whole optimization run. BBA converges at constraint minima with a 0.00% optimality gap [76]. As a result, the Boolean quadratic polynomial problem is solved globally when predetermined stopping and feasibility criteria are fully satisfied [11]. The log file shows that the lower bound closes the absolute gap to ensure that a global solution point is derived [77].

Table A14. 0/1 quadratic constraint polynomial Solver Log File.

```
+ Solver chosen: BMIBNB
+ Processing objective function
+ Processing constraints
+ Branch and bound started
* Starting YALMIP global branch and bound.
|   #3 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|   #4 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|   #5 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|   #6 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|   #7 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|   #8 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|   #9 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|   #10 |  Equality constraint (polynomial) 1 × 1 |     1 to 1 |
|   #11 |  Equality constraint (bilinear) 1 × 1 |       1 to 1 |
|   #12 |  Equality constraint (polynomial) 1 × 1 |     1 to 1 |
|   #13 |  Equality constraint (bilinear) 1 × 1 |       1 to 1 |
|   #14 |  Equality constraint (polynomial) 1 × 1 |     1 to 1 |
|   #15 |  Equality constraint (polynomial) 1 × 1 |     1 to 1 |
|   #16 |  Equality constraint (polynomial) 1 × 1 |     1 to 1 |
|   #17 |  Equality constraint (polynomial) 1 × 1 |     1 to 1 |
```

Table A14. *Cont.*

```
|  #18 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #19 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #20 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #21 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #22 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #23 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #24 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #25 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #26 |  Equality constraint (bilinear) 1 × 1 |     1 to 1 |
|  #27 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #28 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #29 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #30 |  Equality constraint (polynomial) 1 × 1 |   1 to 1 |
|  #31 |       Binary constraint |        |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
* Starting YALMIP global branch and bound.
* Upper solver:   fmincon
* Lower solver:   GUROBI
* LP solver:   GUROBI
* -Extracting bounds from model
* -Performing root-node bound propagation
* -- Calling upper solver + Calling FMINCON (no solution found)
* -Branch-variables: 175
* -More root-node bound-propagation
* -Performing LP-based bound-propagation
* -And some more root-node bound-propagation
* Starting the B AND B process
Node   Upper   Gap(%)       Lower       Open     Time
+ Calling GUROBI
+ Calling GUROBI
+ Calling FMINCON
1:   1.00000E+01   0.00   1.00000E+01   2   4 s Solution found by heuristics
+ Calling GUROBI
2:   1.00000E+01   0.00   1.00000E+01   0 4 s Poor lower bound |   Pruned stack based on new
upper bound
* Finished. Cost: 10 (lower bound: 10, relative gap 0%)
* Termination with all nodes pruned
* Timing: 9% spent in upper solver (2 problems solved)
*   9% spent in lower solver (2 problems solved)
*   51% spent in LP-based domain reduction (350 problems solved)
*   1% spent in upper heuristics (1 candidates tried)
Quadratic scalar (real, homogeneous, 30 variables)
Current value: 10
Coefficients range: 1 to 1
```

Table A15 illustrates the PMU positioning sites derived by optimizing the polynomial problem. The plot diagram is illustrated in Figure A2.

Table A15. The PMU placement sites with the best objective value and a SORI.

Optimal PMU Placement Sites
1, 5, 8, 10, 11, 12, 19, 23, 26, 29
Optimal number of PMUs: 10 SORI: 35 best function value: 10

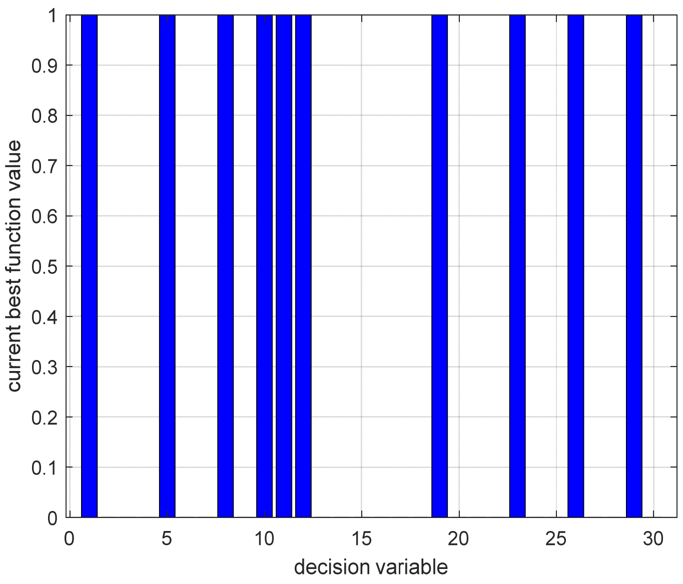

Figure A2. Optimal PMU positioning sites under complete observability.

This optimization is executed by adjusting tuning parameters to optimize an objective function, either in minimization or maximization problems. Tightening the tolerances permits us to optimize more extensively, and this illustrates that the solution point discovered at the final root node certainly was the global solution, and the culprit to close the gap was the lower bound on the objective function [76].

As the log files show, the iterative process is terminated at two root nodes, while the absolute gap is equal to zero and the relative gap is also zero [9–11].

The objective function value is given with a standard optimality criterion that was found to be equal to zero. As a result, a globally optimal solution is attained [9–11].

III. Semi-Definite Programming Model

The IEEE-30 bus system is used as a case study to show the applicability of YALMIP BBA to the 0/1 SDP problem [80]. Table A16 shows the output through a cutting plane for MISDP based on MILP, whereas Table A17 shows the PMU placement sites.

The CUTSDP routine implements a cutting-plane strategy, relaxes the integer decision variables, executes the BBA, and solves conic optimization problems at every node [78].

The conic constraints are relaxed to a linear constraint function; therefore, mixed-integer programming is solved towards optimality [78].

The identified optimality and stopping criterion gap achieved give a clear indication that the SDP problem can be solved exactly [78]. So, the criterion of suboptimality is fully satisfied, a zero-gap tolerance is achieved, and a global solution is achieved [9–11,70–79].

Table A16. 0/1 SDP CUTSDP n Solver Log File.

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
| ID |           Constraint |  Coefficient range |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
| #1 |    Matrix inequality (integer) 30 × 30 |     0.009 to 8 |
| #2 |    Binary constraint (integer) |        |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+ Solver chosen: CUTSDP
+ Processing objective function
+ Processing constraints
+ Cutting plane solver started* Starting YALMIP cutting plane for MISDP based on MILP
* Lower solver:   GUROBI
* Max iterations:   Inf
* Max time:         3600
Node   Phase         Cone infeas   Integrality infeas Lower bound Upper bound LP cuts Elapsed
time
   1: Continuous    0.000E+00    0.000E+00    1.000E+01        Inf          29        0.1
   2: Integer       0.000E+00    0.000E+00    1.000E+01     1.000E+01       30        0.1
Optimal number of PMUs: 10
SORI: 48
best function value: 10
```

Table A17. The PMU placement sites with the best objective value indexed by a SORI.

Optimal PMU Positioning Sites
1, 5, 6, 9, 10, 12, 15, 19, 25, 27

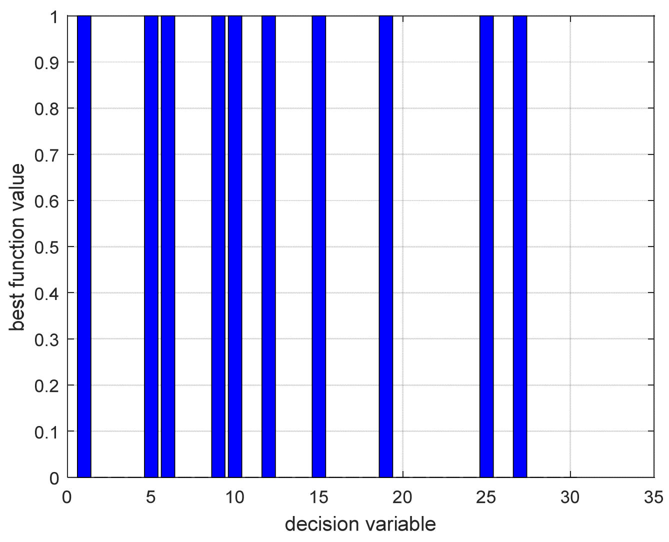

Figure A3. Optimal PMU positioning set under complete observability.

As shown, the SDP objective function gap shows the percentage gap between the level of quality of SDP relaxation and the best-known upper bound [69]. The plot diagram is illustrated in Figure A3.

Appendix D

- Aim to maximize the redundancy of synchronized measurements in power system observability

In Appendix D, we report the BBA solver's progress by illustrating the optimization function log files related to the maximization models. Initially, we optimize the 0/1 ILP model, we transform it into a 0/1 SDP model, and finally, we present the LCBO for a representative small-sized power system [80].

Each log points to the type of the optimization model, in addition to the number of design variables and the entire number of constraints in each model. The absolute and relative gap is an adequate optimality condition that leads to a tendency for a globally optimal solution that has been achieved for each maximization model [9–11,70–79].

By counting the upper and lower bounds to obtain the optimum point, the BBA prunes the feasibility search region, helping to recognize the maximum objective function value [11].

The relaxed solution is the upper bound at each root node, and the existing maximum binary integer solution at any root node is the lower bound [9–11].

In a maximization problem, a lower bound on the optimum solution point is delivered by any feasible solution. The current lower bound being the tightest is determined by the price of the best solution we have detected. This is also mentioned as the incumbent solution. In all cases, the absolute and relative gaps are measured to be zero. As a result, the determination to find a global solution is proven by the log files of the routine. The lower bound is the global solution for the maximization problem [9–11].

I. Mixed-Integer Linear Programming Model

The iterative process ends at one root node, as the log files show in Table A18. The plot diagram is illustrated in Figure A4.

Table A18. 0/1 MILP Solver Log File based on GUROBI optimization library.

```
+ Solver chosen: BMIBNB
+ Processing objective function
+ Processing constraints
+ Branch and bound started
* Starting YALMIP global branch and bound.
* -Performing root-node bound propagation
-Calling upper solver + Calling GUROBI (found a solution!)
* -Branch-variables: 0
* -More root-node bound-propagation
* -Performing LP-based bound-propagation
* -And some more root-node bound-propagation
* Starting the B and B process
Node   Upper           Gap(%)     Lower           Open   Time
   1:  −1.73333E+00    0.00       −1.73333E+00    0      0 s Poor lower bound
* Finished. Cost: −1.7333 (lower bound: −1.7333, relative gap 0%)
* Termination with all nodes pruned
* Timing: 14% spent in upper solver (1 problems solved)
```

Table A18. *Cont.*

* 7% spent in lower solver (1 problems solved)
* 1% spent in LP-based domain reduction (0 problems solved)
* 1% spent in upper heuristics (0 candidates tried)
ans =
Optimal PMU locations
2 4 6 9 10 12 15 20 25 27
Optimal number of PMUs: 10
SORI: 52
obj: −1.733333e+00

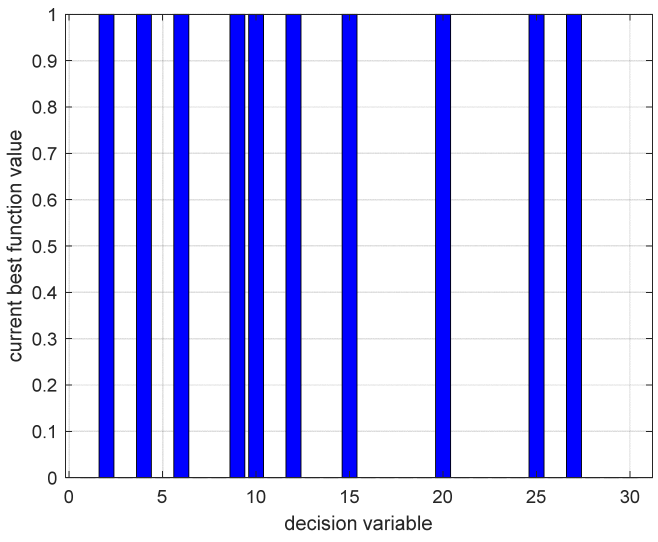

Figure A4. Optimal PMU positioning set undex maximum observability network.

II. Linear Constrained Binary Optimization Model

The iterative process ends up at two root nodes as the log files show in Table A19. The plot diagram is illustrated in Figure A5.

Table A19. 0/1 constraint polynomial-based Solver Log File.

+ Solver chosen: BMIBNB
+ Processing objective function
+ Processing constraints
+ Branch and bound started
* Starting YALMIP global branch and bound.
∣ #3 ∣ Equality constraint (polynomial) 1 × 1 ∣ 1 to 1 ∣
∣ #4 ∣ Equality constraint (polynomial) 1 × 1 ∣ 1 to 1 ∣
∣ #5 ∣ Equality constraint (polynomial) 1 × 1 ∣ 1 to 1 ∣
∣ #6 ∣ Equality constraint (polynomial) 1 × 1 ∣ 1 to 1 ∣
∣ #7 ∣ Equality constraint (polynomial) 1 × 1 ∣ 1 to 1 ∣
∣ #8 ∣ Equality constraint (polynomial) 1 × 1 ∣ 1 to 1 ∣
∣ #9 ∣ Equality constraint (polynomial) 1 × 1 ∣ 1 to 1 ∣

Table A19. *Cont.*

```
|  #10 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #11 |  Equality constraint (bilinear) 1 × 1 |         1 to 1 |
|  #12 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #13 |  Equality constraint (bilinear) 1 × 1 |         1 to 1 |
|  #14 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #15 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #16 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #17 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #18 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #19 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #20 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #21 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #22 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #23 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #24 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #25 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #26 |  Equality constraint (bilinear) 1 × 1 |         1 to 1 |
|  #27 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #28 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #29 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #30 |  Equality constraint (polynomial) 1 × 1 |      1 to 1 |
|  #31 |     Equality constraint 1 × 1 |                 1 to 10 |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
* Starting YALMIP global branch and bound.
* Upper solver:   fmincon
* Lower solver:   GUROBI
* LP solver:   GUROBI
* -Extracting bounds from model
* -Perfoming root-node bound propagation
* -Calling upper solver + Calling FMINCON (no solution found)
* -Branch-variables: 30
* -More root-node bound-propagation
* -Performing LP-based bound-propagation
* -And some more root-node bound-propagation
* Starting the B and B process
   Node      Upper       Gap(%)       Lower       Open       Time
+ Calling GUROBI
+ Calling FMINCON
   1: −1.73333E+00           0.00    −1.73333E+00   2    2 s Solution found by heuristics
+ Calling GUROBI
   2: −1.73333E+00           0.00    −1.73333E+00   0    3 s Poor lower bound |    Pruned stack
based on new upper bound
```

Table A19. *Cont.*

* Finished. Cost: −1.7333 (lower bound: −1.7333, relative gap 0%)
* Termination with all nodes pruned
* Timing: 10% spent in upper solver (2 problems solved)
* 6% spent in lower solver (2 problems solved)
* 19% spent in LP-based domain reduction (60 problems solved)
* 1% spent in upper heuristics (1 candidates tried)
Optimal PMU locations
2 4 6 9 10 12 15 20 25 27
Linear scalar (real, binary, 30 variables)
Current value: −1.7333
Coefficients range: 0.066667 to 0.26667
SORI =
52
ans =
'bmibnb'

The plot diagram is illustrated in Figure A5.

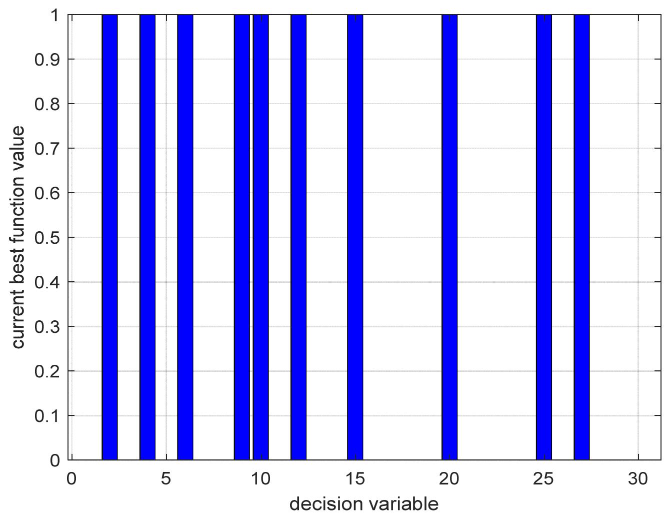

Figure A5. Optimal PMU positioning set under maximum observability condition.

III. Semi Definite Programming

The CUTSDP solver implements a cutting-plane strategy, relaxes the integer decision variables, executes the BBA, and solves conic optimization problems at every node [78].

The conic constraints are relaxed to a linear constraint function; therefore, mixed-integer programming is solved towards optimality [78].

As shown, the SDP objective function gap shows the percentage gap between the level of quality of SDP relaxation and the best-known lower bound [44,78]. Table A21 shows the PMU positioning sites. The plot diagram is illustrated in Figure A6 under the maximum observability condition [22,34,35,39,41,46].

Table A20. YALMIP SDP **CUT**SDP Solver Log File.

```
+ Solver chosen: CUTSDP
+ Processing objective function
+ Processing constraints
+ Cutting plane solver started* Starting YALMIP cutting plane for MISDP based on MILP
* Lower solver: GUROBI
* Max iterations: Inf
* Max time:   3600
Node   Phase    Cone infeas   Integrality infeas Lower bound Upper bound LP cuts Elapsed time
  1:  Continuous    0.000E+00   0.000E+00   −1.733E+00       Inf        29       0.1
  2:  Integer       0.000E+00   0.000E+00   −1.733E+00   −1.733E+00     29       0.1
SORI =
  52
ans =
  ' CUTSDP '
```

Table A21. The Optimal PMU placement sites.

PMU Positioning Sites
2, 4, 6, 9, 10, 12, 15, 20, 25, 27

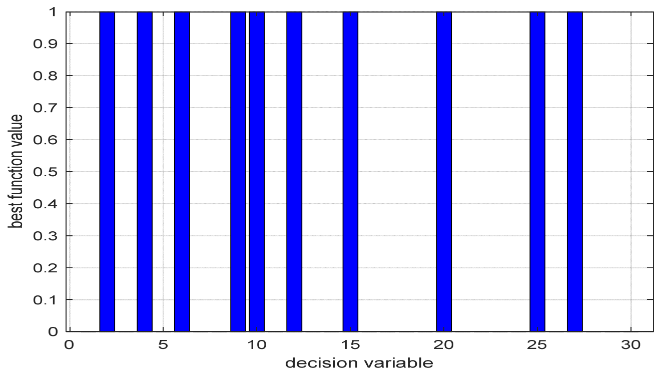

Figure A6. Optimal PMU positioning set under maximum observability condition.

Appendix E

In Appendix E, we report that the binary integer program can give all the solutions for the OPP problem, despite the opposite declared in the bibliography: the ILP can give only one solution [12–18]. Using the Gurobi optimizer routine [74], we can derive a solution pool giving all the solutions. We use the IEEE 57-bus system for this purpose [80]. Firstly, we derive the solutions desired by the minimization problem and then the desired outcome for solving the maximization problem both in a binary integer program [11].

While the GUROBI MILP solver delivers a particular solution point to the ILP model under complete observability [45], the optimizer function allows the user to adjust the tuning parameters [74]. Using the option named PoolSearchMode factor with a proper syntax as params.PoolSearchMode = 2, a solution pool consisting of local solutions is produced within an absolute gap equal to zero [74].

Using the parameter params.PoolSolutions, we set the number of solutions to be delivered by the optimizer routine. The minimization model and the maximization problem are programmed in YALMIP using the above-stated command syntax [69,74].

The Gurobi optimizer engine delivers 3348 optimal solutions with an absolute gap equal to zero, as the Solver Log File based on GUROBI ENGINE illustrates [74].

The past research study was a notation remark that the ILP may be trapped into the local solution point and the measurement redundancy task was not examined [24,36].

To reverse this wrong remark, we revisit the OPP problem through the ILP model programmed in YALMIP. To obtain the global solution, the measurement redundancy task, and all solutions derived from the entire optimization, we go through the GUROBI [74].

Table A22 reveals the truth that integer programming is able to give the exact number of PMU placement sets and all the solutions performed by a suitable MILP solver, despite the opposite referred to in [24,36].

Table A22 illustrates the log files returned by Gurobi performing in the YALMIP environment in an optimization way to find all the solutions by using the solution pool option of Gurobi needed to cover the network observability [74].

Table A22 illustrates the first optimal solution derived by GUROBI with SORI equal to 64, whereas Figure A7 illustrates the plot showing the PMU positioning sites. Table A23 illustrates the first 20 numbers of PMUs; among them is the solution with the maximum SORI [46].

Table A24 illustrates the PMU placement locations with maximum observability indicator. As it is shown, the solver finds the optimal solution with maximum SORI [22]. Table A25 illustrates the GUROBI ENGINE **So**lver Log File for maximization problem.

Table A26 illustrates all the solutions with the maximum observability indicator derived by the GUROBI optimizer engine inside an absolute gap equal to zero [74].

Table A22. YALMIP MILP GUROBI Solver Log File for minimization problem.

```
Gurobi Optimizer version 10.0.0 build v10.0.0rc2 (win64)
Optimize a model with 57 rows, 57 columns and 213 nonzeros
Variable types: 0 continuous, 57 integer (57 binary)
Found heuristic solution: objective 20.0000000
Presolve removed 6 rows and 0 columns
Presolve time: 0.00 s
Presolved: 51 rows, 57 columns, 184 nonzeros
Variable types: 0 continuous, 57 integer (57 binary)
Root relaxation presolved: 51 rows, 57 columns, 184 nonzeros
Root relaxation: objective 1.600000e+01, 69 iterations, 0.01 s (0.00 work units)
   Nodes    |    Current Node    |    Objective Bounds    |    Work
Expl Unexpl |  Obj  Depth IntInf |  Incumbent    BestBd   Gap  |  It/Node Time
   0    0   16.00000   0   20    20.00000    16.00000   20.0%    -    0 s
H  0    0              19.0000000    16.00000 15.8%    -    0 s
H  0    0              17.0000000    16.00000 5.88%    -    0 s
   0    0   16.50000   0   13 17.00000    16.50000 2.94%    -    0 s
Optimal solution found at node 0 - now completing solution pool...
   Nodes    |    Current Node    |    Pool Obj. Bounds    |    Work
            |        |   Worst          |
   Expl Unexpl |  Obj  Depth IntInf |  Incumbent    BestBd   Gap  |  It/Node Time
     0    0   16.50000   0   17          - 16.50000 -    -    0 s
     0    0          - 0          - 17.00000   -    - 0 s
     0    0          - 0          - 17.00000   - -   0 s
     0    0          - 0          - 17.00000   - -   0 s
     0    2          - 0          - 17.00000   - -   0 s
Cutting planes:
   Gomory: 1
   Zero half: 4
Explored 78447 nodes (95209 simplex iterations) in 3.25 s (0.55 work units)
Thread count was four (of four available processors)
Solution count 3348: 17 17 17 ... 17
Optimal solution found (tolerance 1.00e−04)
Best objective 1.700000000000e+01, best bound 1.700000000000e+01, gap 0.0000%
Elapsed time is 3.317972 s.
Optimal objective: 1.700000e+01
```

Table A23. The Optimal PMU placement sites.

PMU Placement Set
1, 4, 7, 10, 20, 23, 25, 26, 29, 32, 36, 39, 41, 44, 46, 49, 54

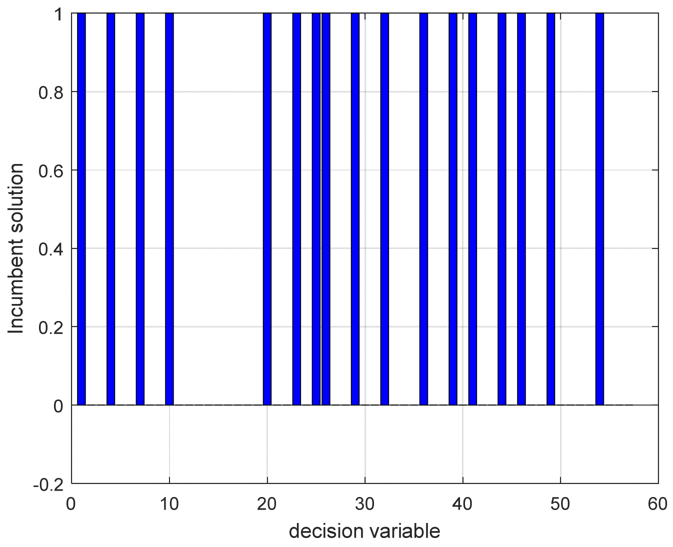

Figure A7. Optimal PMU positioning set under a SORI index equal to 64.

Table A24. Optimal PMU placement locations indexed by a SORI.

PMU Placement Locations under Complete Observability Scenario	SORI
1, 4, 7, 10, 20, 23, 25, 26, 29, 32, 36, 39, 41, 44, 46, 49, 54	64
2, 6, 12, 19, 22, 25, 27, 32, 36, 38, 41, 45, 46, 50, 52, 55, 57	64
1, 4, 7, 9, 20, 22, 25, 27, 32, 36, 38, 41, 45, 46, 50, 53, 57	68
1, 6, 7, 9, 15, 19, 22, 25, 27, 32, 36, 38, 41, 47, 50, 53, 57	71
1, 6, 9, 15, 19, 22, 25, 27, 32, 36, 38, 41, 47, 50, 52, 55, 57	70
2, 6, 12, 15, 19, 22, 25, 27, 32, 36, 38, 41, 47, 50, 52, 55, 57	67
2, 6, 12, 19, 22, 25, 26, 29, 32, 36, 38, 41, 45, 46, 50, 54, 57	65
2, 6, 12, 19, 22, 25, 27, 29, 32, 36, 38, 41, 45, 46, 50, 54, 57	65
2, 6, 12, 19, 22, 25, 27, 29, 32, 36, 41, 45, 46, 49, 50, 54, 57	64
2, 6, 12, 19, 22, 25, 27, 29, 32, 36, 38, 41, 45, 46, 51, 54, 57	65
2, 6, 12, 19, 22, 25, 27, 29, 32, 36, 38, 39, 41, 45, 46, 50, 54	65
2, 6, 12, 19, 22, 25, 27, 32, 36, 38, 41, 45, 46, 51, 52, 55, 57	64
2, 6, 12, 19, 22, 25, 27, 32, 36, 38, 39, 41, 45, 46, 50, 52, 55	64
2, 6, 12, 19, 22, 25, 27, 32, 36, 41, 45, 46, 49, 50, 52, 55, 57	63
1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 41, 46, 50, 53, 57	**72**
1, 5, 7, 9, 15, 19, 22, 25, 27, 32, 36, 38, 41, 47, 50, 53, 57	69
2, 6, 12, 19, 22, 25, 26, 29, 32, 36, 41, 45, 46, 49, 50, 54, 57	64
2, 6, 12, 19, 22, 25, 27, 32, 36, 38, 39, 41, 45, 46, 51, 52, 54	64
2, 6, 12, 19, 22, 25, 26, 29, 32, 36, 38, 41, 45, 46, 51, 54, 57	65
2, 6, 12, 19, 22, 26, 29, 30, 32, 36, 38, 41, 45, 46, 51, 54, 57	65

Table A25. GUROBI ENGINE Solver Log File for maximization problem.

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|   ID |   Constraint |   Coefficient range |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|   #1 |   Element-wise inequality 57 × 1 |        1 to 1 |
|   #2 |   Equality constraint 1 × 1 |        1 to 17 |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Set parameter PoolSolutions to value 24
Set parameter PoolSearchMode to value 2
Optimize a model with 58 rows, 57 columns and 270 nonzeros
Variable types: 0 continuous, 57 integer (57 binary)
Presolve removed 6 rows and 0 columns
Presolve time: 0.00 s
Presolved: 52 rows, 57 columns, 241 nonzeros
Root relaxation: objective −1.271930e+00, 38 iterations, 0.00 s (0.00 work units)
   Nodes    |    Current Node    |    Objective Bounds    |    Work
   Expl Unexpl  |   Obj Depth IntInf   |   Incumbent BestBd Gap   |   It/Node Time

   0   0   −1.27193   0   10         -      −1.27193        -       -    0 s
H   0   0              −1.1929825       −1.27193   6.62%      -    0 s
H   0   0              −1.2456140       −1.27193   2.11%      -    0 s
H   0   0              −1.2631579       −1.27193   0.69%      -    0 s
   0   0     -      0   −1.26316        −1.26316   0.00%      -    0 s
Optimal solution found at node 0 - now completing solution pool...
   Nodes    |    Current Node    |    Pool Obj. Bounds    |    Work
         |         |  Worst       |
Expl Unexpl   |   Obj Depth IntInf   |   Incumbent BestBd Gap   |   It/Node Time

     0   0            -  0           -      −1.26316      -  -   0 s
     0   0            -  0           -      −1.26316      -  -   0 s
     0   2            -  0           -      −1.26316      -  -   0 s
Cutting planes:
  Zero half: 1
Explored 299 nodes (279 simplex iterations) in 0.18 s (0.00 work units)
Solution count 24: −1.26316 −1.26316 −1.26316 . . . −1.26316
No other solutions better than −1.26316
Optimal solution found (tolerance 1.00e−04)
Best objective −1.263157894737e+00, best bound −1.263157894737e+00, gap 0.0000%
```

Table A26. Optimal PMU placement locations with maximum observability index (SORI) equal to 72.

PMU Placement Locations under Maximum Observability Index
1, 4, 6, 9, 15, 20, 24, 28, 30, 32, 36, 38, 41, 46, 51, 53, 57
1, 4, 6, 9, 15, 20, 24, 28, 30, 32, 36, 38, 41, 46, 50, 53, 57
1, 4, 6, 9, 15, 20, 24, 28, 30, 32, 36, 38, 39, 41, 46, 51, 53
1, 4, 6, 9, 15, 20, 24, 28, 30, 32, 36, 38, 39, 41, 46, 50, 53
1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 41, 46, 51, 53, 57
1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 41, 46, 51, 53, 57
1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 41, 47, 51, 53, 57
1, 4, 6, 9, 15, 20, 24, 28, 30, 32, 36, 38, 41, 47, 51, 53, 57
1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 41, 47, 51, 53, 57
1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 41, 47, 50, 53, 57
1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 41, 46, 50, 53, 57
1, 4, 6, 9, 15, 20, 24, 28, 30, 32, 36, 38, 41, 47, 50, 53, 57
1, 4, 6, 9, 15, 20, 24, 28, 30, 32, 36, 38, 39, 41, 47, 51, 53
1, 4, 6, 9, 15, 20, 24, 28, 30, 32, 36, 38, 39, 41, 47, 50, 53
1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 41, 46, 50, 53, 57
1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 39, 41, 46, 51, 53

Table A26. *Cont.*

PMU Placement Locations under Maximum Observability Index
1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 39, 41, 47, 51, 53
1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 39, 41, 47, 50, 53
1, 4, 6, 9, 15, 20, 24, 25, 28, 32, 36, 38, 39, 41, 46, 50, 53
1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 41, 47, 50, 53, 57
1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 39, 41, 46, 51, 53
1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 39, 41, 47, 51, 53
1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 39, 41, 47, 50, 53
1, 4, 6, 9, 15, 20, 24, 28, 31, 32, 36, 38, 39, 41, 46, 50, 5 3
best function value: −1.263158e+00
Optimal number of PMUs: 17
SORI: 72

References

1. Phadke, A.G.; Thorp, J.S. *Synchronized Phasor Measurements and Their Applications*, 2nd ed.; Springer: New York, NY, USA, 2017. [CrossRef]
2. Mohamed, E. *El-Hawary Advances in Electric Power and Energy: Static State Estimation*; John Wiley & Sons: Hoboken, NJ, USA, 2020.
3. Cyman, J.; Raczek, J. Application of Doubly Connected Dominating Sets to Safe Rectangular Smart Grids. *Energies* **2022**, *15*, 2969. [CrossRef]
4. Brimkov, B.; Mikesell, D.; Smith, L. Connected power domination in graphs. *J. Comb. Optim.* **2019**, *38*, 292–315. [CrossRef]
5. Haynes, T.W.; Hedetniemi, S.M.; Hedetniemi, S.T.; Henning, M.A. Domination in graphs applied to electric power networks. *SIAM J. Discret. Math.* **2002**, *15*, 519–529. [CrossRef]
6. Diestel, R. *Graph Theory*; Springer: Berlin, Germany, 2017; Volume 173, pp. 59–172.
7. Narsingh, D. *Graph Theory with Applications to Engineering and Computer Science*; Prentice-Hall Inc.: Mineola, NY, USA, 1974.
8. Christofides, N. *Graph Theory. An Algorithmic Approach'*; Academic Press Inc.: Cambridge, MA, USA, 1975.
9. Arora, J.S. *Introduction to Optimum Design*, 4th ed.; Elsevier Inc.: Amsterdam, The Netherlands, 2016.
10. Williams, H.P. *Model Building in Mathematical Programming*; John Wiley & Sons: New York, NY, USA, 2013.
11. Karlof, J. *Integer Programming and Practice*, 1st ed.; CRC Press Taylor & Francis Group, 6000 Broken Sound Parkway NW, Suite 300: Boca Raton, FL, USA, 2006.
12. Ahmed, M.M.; Amjad, M.; Qureshi, M.A.; Imran, K.; Haider, Z.M.; Khan, M.O. A Critical Review of State-of-the-Art Optimal PMU Placement Techniques. *Energies* **2022**, *15*, 2125. [CrossRef]
13. Biswal, C.; Sahu, B.K.; Mishra, M.; Rout, P.K. Real-Time Grid Monitoring and Protection: A Comprehensive Survey on the Advantages of Phasor Measurement Units. *Energies* **2023**, *16*, 4054. [CrossRef]
14. Paramo, G.; Bretas, A.; Meyn, S. Research Trends and Applications of PMUs. *Energies* **2022**, *15*, 5329. [CrossRef]
15. Menezes, T.S.; Barra, P.H.A.; Dizioli, F.A.S.; Lacerda, V.A.; Fernandes, R.A.S.; Coury, D.V. A Survey on the Application of Phasor Measurement Units to the Protection of Transmission and Smart Distribution Systems. *Electr. Power Compon. Syst.* **2023**, *1*, 1–18. [CrossRef]
16. Joshi, P.M.; Verma, H.K. Synchrophasor measurement applications and optimal PMU placement: A review. *Electr. Power Syst. Res.* **2021**, *199*, 107428.
17. Mohanta, D.K.; Murthy, C.; Sinha Roy, D. A Brief Review of Phasor Measurement Units as Sensors for Smart Grid. *Electr. Power Compon. Syst.* **2016**, *44*, 411–425.
18. Johnson, T.; Moger, T. A critical review of methods for optimal placement of phasor measurement units. *Int. Trans. Electr. Energy Syst.* **2020**, *31*, e12698.
19. Poirion, P.L.; Toubaline, S.; D'Ambrosio, C.; Liberti, L. The power edge set problem. *Networks* **2016**, *68*, 104–120. [CrossRef]
20. Bei, X.; Yoon, Y.J.; Abur, A. *Optimal Placement and Utilization of Phasor Measurements for State Estimation*; PSERC Publication: Tempe, AZ, USA, 2005; pp. 1–6.
21. Xu, B.; Abur, A. Observability Analysis and Measurement Placement for Systems with PMUs. In Proceedings of the IEEE PES Power Systems Conference and Exposition, New York, NY, USA, 10–13 October 2004; pp. 2–5.
22. Dua, D.; Dambhare, S.; Gajbhiye, R.K.; Soman, S.A. Optimal Multistage Scheduling of PMU Placement: An ILP Approach. *IEEE Trans. Power Deliv.* **2008**, *23*, 1812–1820. [CrossRef]
23. Gou, B. Generalized integer linear programming formulation for optimal PMU placement. *IEEE Trans. Power Syst.* **2008**, *23*, 1099–1104. [CrossRef]
24. Chakrabarti, S.; Kyriakides, E.; Eliades, D.G. Placement of synchronized measurements for power system observability. *IEEE Trans. Power Deliv.* **2009**, *24*, 12–19. [CrossRef]
25. Bečejac, V.; Stefanov, P. Groebner bases algorithm for optimal PMU placement. *Int. J. Electr. Power Energy Syst.* **2020**, *115*, 105427. [CrossRef]

26. Korres, G.N.; Manousakis, N.M.; Xygkis, T.C.; Lofberg, J. Optimal phasor measurement unit placement for numerical observability in the presence of conventional measurement using semi-definite programming. *IET Gener. Transm. Distrib.* **2015**, *9*, 2427–2436. [CrossRef]
27. Almunif, A.; Fan, L. DC State Estimation Model-Based Mixed Integer Semidefinite Programming for Optimal PMU Placement. In Proceedings of the 2018 North American Power Symposium (NAPS), Fargo, ND, USA, 9–11 September 2018; pp. 1–6.
28. Manousakis, N.M.; Korres, G.N. Optimal Allocation of Phasor Measurement Units Considering Various Contingencies and Measurement Redundancy. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 3403–3411. [CrossRef]
29. Hyacinth, L.R.; Gomathi, V. Optimal pmu placement technique to maximize measurement redundancy based on closed neighbourhood search. *Energies* **2021**, *14*, 4782. [CrossRef]
30. Xie, N.; Torelli, F.; Bompard, E.; Vaccaro, A. A graph theory based methodology for optimal PMUs placement and multiarea power system state estimation. *Electr. Power Syst. Res.* **2015**, *119*, 25–33. [CrossRef]
31. Müller, H.H.; Castro, C.A. Genetic algorithm-based phasor measurement unit placement method considering observability and security criteria. *IET Gener. Transm. Distrib.* **2016**, *10*, 270–280. [CrossRef]
32. Theodorakatos, N.P. Optimal Phasor Measurement Unit Placement for Numerical Observability Using Branch-and-Bound and a Binary-Coded Genetic Algorithm. *Electr. Power Compon. Syst.* **2019**, *47*, 357–371. [CrossRef]
33. Dalali, M.; Karegar, H.K. Optimal PMU placement for full observability of the power network with maximum redundancy using modified binary cuckoo optimization algorithm. *IET Gener. Transm. Distrib.* **2016**, *10*, 2817–2824. [CrossRef]
34. Singh, S.P.; Singh, S.P. A Multi-objective PMU Placement Method in Power System via Binary Gravitational Search Algorithm. *Electr. Power Compon. Syst.* **2017**, *45*, 1832–1845. [CrossRef]
35. Rahman, N.H.A.; Zobaa, A.F. Optimal PMU placement using topology transformation method in power systems. *J. Adv. Res.* **2016**, *7*, 625–634. [CrossRef]
36. Chakrabarti, S.; Venayagamoorthy, G.K.; Kyriakides, E. PMU placement for power system observability using binary particle swarm optimization. In Proceedings of the 2008 Australasian Universities Power Engineering Conference, Sydney, NSW, Australia, 14–17 December 2008; pp. 1–5.
37. Maji, T.K.; Acharjee, P. Multiple Solutions of Optimal PMU Placement Using Exponential Binary PSO Algorithm for Smart Grid Applications. *IEEE Trans. Ind. Appl.* **2017**, *53*, 2550–2559. [CrossRef]
38. Johnson, T.; Moger, T. Security-constrained optimal placement of PMUs using Crow Search Algorithm. *Appl. Soft Comput.* **2022**, *128*, 109472. [CrossRef]
39. Ramasamy, S.; Koodalsamy, B.; Koodalsamy, C.; Veerayan, M.B. Realistic method for placement of phasor measurement units through optimization problem formulation with conflicting objectives. *Electr. Power Compon. Syst.* **2021**, *49*, 474–487. [CrossRef]
40. Logeshwari, V.; Abirami, M.; Subramanian, S.; Manoharan, H. Multi-objective precise phasor measurement locations to assess small-signal stability using dingo optimizer. *Optim. Control. Appl. Methods* **2023**. [CrossRef]
41. Xia, N.; Gooi, H.B.; Chen, S.X.; Wang, M.Q. Redundancy based PMU placement in state estimation. *Sustain. Energy Grids Netw.* **2015**, *2*, 23–31. [CrossRef]
42. Yang, X.S. *Engineering Optimization: An Introduction with Metaheuristic Applications*; Wiley: Hoboken, NJ, USA, 2010.
43. Manousakis, N.M.; Korres, G.N. A Weighted Least Squares Algorithm for optimal PMU placement. *IEEE Trans. Power Syst.* **2013**, *28*, 3499–3500. [CrossRef]
44. Chinneck, J.W. *Feasibility and Infeasibility in Optimization: Algorithms and Computational Methods*; International Series in Operations Research & Management Science; Springer: Cham, Switzerland, 2008.
45. Theodorakatos, N.P.; Lytras, M.; Babu, R. Towards Smart Energy Grids: A Box-Constrained Nonlinear Underdetermined Model for Power System Observability Using Recursive Quadratic Programming. *Energies* **2020**, *13*, 1724. [CrossRef]
46. Theodorakatos, N.P.; Lytras, M.; Babu, R. A Generalized Pattern Search Algorithm Methodology for solving an Under-Determined System of Equality Constraints to achieve Power System Observability using Synchrophasors. *J. Phys. Conf. Ser.* **2021**, *2090*, 012125. [CrossRef]
47. Patel, C.D.; Tailor, T.K.; Shah, S.S.; Shrivastava, S.H. An approach for economic design of wide area monitoring system by co-optimizing phasor measurement unit placement and associated communication infrastructure. *Int. Trans. Electr. Energy Syst.* **2021**, *31*, e12977. [CrossRef]
48. Baba, M.; Nor, N.B.; Aman Sheikh, M.; Irfan, M.; Tahir, M. A Strategic and Significant Method for the Optimal Placement of Phasor Measurement Unit for Power System Network. *Symmetry* **2020**, *12*, 1174. [CrossRef]
49. Baba, M.; Nor, N.B.M.; Sheikh, M.A.; Baba, A.M.; Irfan, M.; Glowacz, A.; Kozik, J.; Kumar, A. Optimization of Phasor Measurement Unit Placement Using Several Proposed Case Factors for Power Network Monitoring. *Energies* **2021**, *14*, 5596. [CrossRef]
50. Bonavolontà, F.; Caragallo, V.; Fatica, A.; Liccardo, A.; Masone, A.; Sterle, C. Optimization of IEDs Position in MV Smart Grids through Integer Linear Programming. *Energies* **2021**, *14*, 3346. [CrossRef]
51. Almunif, A.; Fan, L. Mixed integer linear programming and nonlinear programming for optimal PMU placement. In Proceedings of the 2017 North American Power Symposium (NAPS), Morgantown, WV, USA, 17–19 September 2017; pp. 1–6.
52. Fortuna, L.; Buscarino, A. Nonlinear Technologies in Advanced Power Systems: Analysis and Control. *Energies* **2022**, *15*, 5167. [CrossRef]
53. Livanos, N.-A.I.; Hammal, S.; Giamarelos, N.; Alifragkis, V.; Psomopoulos, C.S.; Zois, E.N. OpenEdgePMU: An Open PMU Architecture with Edge Processing for Future Resilient Smart Grids. *Energies* **2023**, *16*, 2756. [CrossRef]

54. Zhang, Y.; Li, T.; Na, G.; Li, G.; Li, Y. Optimized Extreme Learning Machine for Power System Transient Stability Prediction Using Synchrophasors. *Math. Probl. Eng.* **2015**, *2015*, 529724. [CrossRef]
55. Arefin, A.A.; Baba, M.; Singh, N.S.S.; Nor, N.B.M.; Sheikh, M.A.; Kannan, R.; Abro, G.E.M.; Mathur, N. Review of the Techniques of the Data Analytics and Islanding Detection of Distribution Systems Using Phasor Measurement Unit Data. *Electronics* **2022**, *11*, 2967. [CrossRef]
56. Saldaña-González, A.E.; Sumper, A.; Aragüés-Peñalba, M.; Smolnikar, M. Advanced Distribution Measurement Technologies and Data Applications for Smart Grids: A Review. *Energies* **2020**, *13*, 3730. [CrossRef]
57. Dusabimana, E.; Yoon, S.-G. A Survey on the Micro-Phasor Measurement Unit in Distribution Networks. *Electronics* **2020**, *9*, 305. [CrossRef]
58. Shahriar, M.S.; Habiballah, I.O.; Hussein, H. Optimization of Phasor Measurement Unit (PMU) Placement in Supervisory Control and Data Acquisition (SCADA)-Based Power System for Better State-Estimation Performance. *Energies* **2018**, *11*, 570. [CrossRef]
59. Manousakis, N.M.; Korres, G.N. A hybrid power system state estimator using synchronized and unsynchronized sensors. *Int. Trans. Electr. Energy Syst.* **2018**, *28*, e2580. [CrossRef]
60. Zhao, J.; Netto, M.; Mili, L. A Robust Iterated Extended Kalman Filter for Power System Dynamic State Estimation. *IEEE Trans. Power Syst.* **2017**, *32*, 3205–3216. [CrossRef]
61. Zhao, J.; Mili, L. A Robust Generalized-Maximum Likelihood Unscented Kalman Filter for Power System Dynamic State Estimation. *IEEE J. Sel. Top. Signal Process.* **2018**, *12*, 578–592. [CrossRef]
62. Aljabrine, A.A.; Smadi, A.A.; Chakhchoukh, Y.; Johnson, B.K.; Lei, H. Resiliency Improvement of an AC/DC Power Grid with Embedded LCC-HVDC Using Robust Power System State Estimation. *Energies* **2021**, *14*, 7847. [CrossRef]
63. Smadi, A.A.; Johnson, B.K.; Lei, H.; Aljabrine, A.A. Improving Hybrid Ac/dc Power System Resilience Using Enhanced Hybrid Power State Estimator. In Proceedings of the 2023 IEEE Power & Energy Society General Meeting (PESGM), Orlando, FL, USA, 16–20 July 2023; pp. 1–5. [CrossRef]
64. Eichfelder, G.; Kirst, P.; Meng, L.; Stein, O. A general branch-and-bound framework for continuous global multiobjective optimization. *J. Glob. Optim.* **2021**, *80*, 195–227. [CrossRef]
65. Morrison, D.R.; Jacobson, S.H.; Sauppe, J.J.; Sewell, E.C. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discret. Optim.* **2016**, *19*, 79–102. [CrossRef]
66. De Ita, G.; Bello, P.; Tovar, M. A Branch and Bound Algorithm for Counting Independent Sets on Grid Graphs. *Comput. Sci. Math. Forum* **2023**, *7*, 28. [CrossRef]
67. Su, C.-H.; Wang, J.-Y. A Branch-and-Bound Algorithm for Minimizing the Total Tardiness of Multiple Developers. *Mathematics* **2022**, *10*, 1200. [CrossRef]
68. Lewis, M.; Glover, F. Quadratic unconstrained binary optimization problem preprocessing: Theory and empirical analysis. *Networks* **2017**, *70*, 79–97. [CrossRef]
69. Löfberg, J. YALMIP: A toolbox for modeling optimization in MATLAB. In Proceedings of the 2004 IEEE International Conference on Robotics and Automation, New Orleans, LA, USA, 2–4 September 2004; pp. 284–289.
70. The MathWorks Inc. Documentation: Intlinprog. Available online: https://es.mathworks.com/help/optim/ug/intlinprog.html (accessed on 15 September 2023).
71. MathWorks. Fmincon. 2018. Available online: https://www.mathworks.com/help/optim/ug/fmincon.html (accessed on 15 September 2023).
72. Achterberg, T. SCIP: Solving constraint integer programs. *Math. Progr. Comput.* **2009**, *1*, 1–41. [CrossRef]
73. Bestuzheva, K.; Besançon, M.; Chen, W.K.; Chmiela, A.; Donkiewicz, T.; van Doornmalen, J.; Eifler, L.; Gaul, O.; Gamrath, G.; Gleixner, A.; et al. Enabling research through the SCIP optimization suite 8.0. *ACM Trans. Math. Softw.* **2023**, *49*, 1–21. [CrossRef]
74. Gurobi. Available online: http://www.gurobi.com (accessed on 15 July 2023).
75. Shalileh, S. An Effective Partitional Crisp Clustering Method Using Gradient Descent Approach. *Mathematics* **2023**, *11*, 2617. [CrossRef]
76. Available online: https://yalmip.github.io/_posts/tutorials/2016-09-17-globaloptimization/ (accessed on 8 August 2023).
77. Available online: https://YALMIP.github.io/solver/bmibnb/ (accessed on 7 September 2023).
78. Available online: https://yalmip.github.io/solver/cutsdp/ (accessed on 15 September 2023).
79. Parametric Fusion (MOSEK 10.1). Available online: https://www.mosek.com (accessed on 10 October 2023).
80. Available online: https://icseg.iti.illinois.edu/power-cases/ (accessed on 30 September 2023).