Pesquisa Operacional (2021) 41: e231454 p.1-25
doi: 10.1590/0101-7438.2021.041.00231454

Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope
ARTICLES

# A CHEMICAL REACTION OPTIMIZATION ALGORITHM FOR PHASOR MEASUREMENT UNIT PLACEMENT

Aileen Carniel[1*] and Mario Mestria[2]

Received November 26, 2019 / Accepted October 18, 2021

**ABSTRACT.** Optimal Phasor Measurement Unit (PMU) Placement (OPP) aims to reduce equipment costs while maintaining electrical systems' observability. PMU´s synchronized phasor data provide electrical systems' information with high quality and frequency enabling the implementation of faster than real-time state estimators. We applied Chemical Reaction Optimization (CRO) method at OPP problem, testing it successfully at IEEE power system databases (14, 30, 57 and 118 bars) through two distinct models. The relationships between CRO parameters and the occurrence of elementary reactions was exploited, achieving better results through specific reactions. Due to problems similarity and larger scales, the software was adapted to Beasley OR-Library Set Covering (SC) problems. In the process to achieve GAPs smaller than 10% for some files, we tried out dedicated local searches, disturbance recurrence limits and stop condition changes. However, we suggest continuing to evaluate CRO method adherence to SC problem using different data structures to decrease computational times.

**Keywords**: Combinatorial Optimization, Phasor Measurement Unit, Observability, Smart Grids, Chemical Reaction Optimization.

## 1 INTRODUCTION

Currently, power systems operate tend to operate in stressful situations, since worldwide energy demand steadily increases. The availability and reliability of electricity networks have a profound impact on society, especially on industry, where energy failures and instabilities might cause financial losses.

A traditional power grid consists basically of power plants, transmission, and distribution systems with unidirectional energy flow, from generation units to consumers. At the end of 20th

*Corresponding author
[1] Instituto Federal do Espírito Santo, Campus Vitória, Avenida Vitória, 1729, Jucutuquara, 29040-780 Vitória-ES, Brazil – E-mail: aileenzc@gmail.com – https://orcid.org/0000-0001-8394-5007
[2] Instituto Federal do Espírito Santo, Campus Vitória, Avenida Vitória, 1729, Jucutuquara, 29040-780 Vitória-ES, Brazil – E-mail: mmestria@uol.com.br – https://orcid.org/0000-0001-8283-0806

century, Supervisory Control and Data Acquisition Systems (SCADA) were implemented to remotely monitor and control the substation equipment. These systems receive energy data such as voltage, current, and active and reactive power data from the RTUs (Remote Terminal Units) at a frequency of 2 to 30 seconds. The low communication rate restricts SCADA to quasi-stationary operating conditions with absence of transient events.

The implementation of high precision state estimators was possible after the development of the synchrophasor technology in the 1980s. A GPS (Global Position System) was inserted at PMUs, which are synchronous bi-directional measurement units capable of transmitting module and phasor data about voltage, current, and active and reactive power. The PMUs communication rate approximates 60 measurements per second, enabling control systems that are faster than real-time, or to human perception, therefore preventing transient instabilities. In order to be applied to electrical systems state estimators must be fully observable. Two different approaches might be used for the observability analysis, numerical and topological. The latter is simpler and, consequently, it is more frequently used.

Due to the bidirectional energy metering, PMUs and other smart meters integrate renewable energy sources, microgrids, and distributed generation to electrical systems enabling a smart grid. Smart grid technology not only inserts new energy sources to power systems, but also enhances production, distribution, and energy consumption. It maximizes equipment usage, increases the availability of systems, and reduces energy loss due to real-time monitoring and control systems, making the operational activities more efficient and safer. Distributed energy resources and additional innovations associated to smart grids allow enhancing the networks by changing their topologies to minimize the total losses due to electrical resistances in the lines and complementary equipment (Cavalheiro, Vergílio & Lyra, 2018).

In distribution networks, smart grids are a part of smart cities. Smart cities are composed of intelligent systems that interact with energy, materials, services, and the economy. Their focus is in the economic development of the city and in the improvement of life quality. Intelligence is inserted through infrastructure, strategies of service usage, and through data collection for urban planning and management. Usually the smart grids implementation takes place in 3 stages:

1. Adding intelligence to electric power system through network automation and monitoring by meters in a reliable communication using PMUs;

2. Replacing electromechanical meters by intelligent electronic ones, which allows bidirectional measurement;

3. Adding intelligence to consumer centers through intelligent equipment that optimize energy consumption, uses renewable energy sources, and energy storage, including electric vehicle charge stations.

The RTUs replacement by PMUs at transmission and distribution networks is affected by the high cost of PMUs and telecommunication equipment. The price of PMUs is determined by the functionalities provided by them and by the number of channels. Mohammadi et al. (2016) esti-

mated the cost of a PMU with two measurement channels in US$ 40,000, with a US$ 4,000 added for each additional channel. Also, a switch and 1 km of optic fiber, required for the installation, are both estimated in US$ 4,000.

Such financial constraints elicited the search for the optimization of electrical meter placement, called OPP (Optimal PMU Placement), in order to allocate the minimum amount of metering devices as PMUs. According to (Liao, Hsieh, Guo, Liu & Chu, 2015), OPP aims to simultaneously minimize the number of PMUs and to ensure the complete observability of the whole power grid.

NIST (2012) describes that meter devices in a smart grid generate a negligible amount of load (amount of data being carried) to the network, and moreover, a smart grid plan must ensure that all meter devices can be keep a reliable communication. In our case, we can evaluate the need of one or more PMUs.

The OPP issue has been studied since the 1990s and has already been solved through different optimization techniques, from heuristics to mathematical methods, hybrid models, and matrix reduction methods. Currently, there is a trend for heuristic optimization methods and, in addition to that, as stated by (Zhou, Centeno, Phadke, Hu, Novosel & Volskis, 2008) OPP is classified as a NP-hard problem, for which heuristic methods can produce robust solutions in faster computational time.

As stated before, OPP can be considered a Set Covering Problem, since each electrical bus that is selected to receive a PMU installation covers the measurement of a certain number of adjacent buses. Hence, when selecting a group of buses to install PMUs, a group of subsets is selected in order to cover the whole search space, in this case, the electrical system.

This paper aims to solve the OPP problem by CRO method, applying topological observability to optimize IEEE power test systems with a minimum number of PMUs, details in the section 3. A model for IEEE power systems was implemented and best known solutions were reached with different CRO parameterization. More than 3500 combinations were compiled. CRO parameters behavior and the relationships between CRO parameters and the occurrence of elementary reactions was exploited, achieving better results through specific reactions.

The tests in IEEE power system databases through two distinct models with CRO method reached optimal values. Thus, due to the similarity between the OPP and SCP, the algorithm to solve the OPP problem was adapted to solve SCP for larger scales using Beasley OR-Library instances. So, this other algorithm was also tested at OR-library for large size instances available to set covering problems. In the process to achieve GAPs smaller than 10% for some files, we tried out dedicated local searches, disturbance recurrence limits and stop condition changes. However, we suggest continuing to evaluate CRO method adherence to SCP using different data structures to decrease computational times. In summary, the CRO was proposed in this paper for OPP and SCP problems.

This paper is organized as follows: Section 2 presents a literature review about the observability of electrical systems, heuristic algorithms applied to the OPP problem, and the Chemical Reac-

tion Optimization. Section 3 describes the methods employed for the solution of the OPP problem, presenting initial solution restrictions, disturbance procedures, the parameterization process, and computational results. Section 4 shows the method developed for the Set Covering Problem, including local search procedures, initial population restriction groups, disturbance procedures, stopping criterion tests, and other specific adaptations. Finally, we provide conclusions and final remarks in Section 5.

## 2 LITERATURE REVIEW

### 2.1 Electrical System Observability

With the goal of analyzing the observability of an electrical system, an incidence matrix was generated and filled with the system measurement relations. It indicates whether each system bus is observable directly or indirectly (if its data are measured from other measured data).

If the system is completely observable, its state variables can be inferred through a $h(\mathbf{x})$ linear function or a $H$ design matrix that relates the estimated states to the actual measurements. The observability of a system might be analyzed either numerically or topologically.

In topological observability analysis power system is represented by a topological graph. The graph has ***'n'*** number of nodes representing the bar of the network bus and ***'e'*** number of edges representing the branches of the network connecting the bus bars. In topological approach the optimal PMU placement set is searched such that each bus of the network is observable by at least one PMU as described in Roy, Sinha & Pradhan (2012).

The system is said to be numerically observable if $H$ design matrix has full rank (Nazari-Heris & Mohammadi-Ivatloo, 2015). However, $H$ matrix is complex to be set up due to the high magnitude of the electrical system. Therefore, the use of topological observability through defined rules of analysis is preferred, especially when zero injection buses, that do not have energy or load injection, are present in the electrical systems.

In order to define the topological rules, direct measurements are made by the installed PMUs and pseudo-measurements are calculated using the previous ones. At zero injection buses (ZIBs), it is possible to estimate voltage and/or current in one of its adjacent buses through the Kirchhoff's laws. ZIBs are the buses which have neither any generation nor any load. According to Roy, Sinha, and Pradhan (2012), at zero injection buses no current is injected into the system, and this is used as pseudo information to make system observable with a smaller number of PMUs compared to the case when information of ZIBs is not considered.

The observability of the electrical system might be handled as an optimization model restriction or might be inserted as a penalty at the objective function, setting up a multi-target problem. Some works include the measurement of redundancy maximization at the objective function in order to assure the observability of the system in cases of a PMU and/or a branch data acquisition failure. In these cases, at least one direct or indirect redundant measurement is added to each bus.

### 2.2 Optimization algorithms applied to the OPP problem

Lam & Li (2012) claims that all metaheuristics have the same performance when compared through the average of several objective functions, although it is usual for a specific method to fit one class of problems better than others. Pereira et al. (2018) claimed that a metaheuristic is a set of concepts that can be used to define heuristic methods applicable to an extensive set of different problems.

Due to the malleability of electrical networks, lower computational development and execution time is required when compared to mathematical methods. Silva & Mestria (2018) used a metaheuristic method to solve a combinatorial optimization problem and reached results as exact as hybrid methods from literature in terms of accuracy and execution time.

Similarly, several heuristic optimization methods were applied to the OPP problem. Among them are "Particle Swarm Optimization" (PSO), "Simulated Annealing" (SA), "Tabu Search" (TS), "Bacteria Foraging Optimization" (BFO), "Artificial Bees Colony" (ABC), "Cellular Learning Automata" (CLA) and "Differential Evolution" (DE).

Both Mohammadi-Ivatloo (2009) and Bedekar et al. (2011) have solved the OPP problem through Genetic Algorithms (GA), while Hui-Ling et al. (2013) presented an algorithm that combines Minimum Span Tree (MST) with an improved GA, called MST-GA. A binary discrete version of the PSO, based on the behavior of particles at the environment, called Modified Binary Particle Swarm Optimization (BPSO) was used by Hajian et al. (2011) who added a new topological observability rule to maximize the use of existing data.

The study of Su & Chen (2010) included individual costs for each PMU installed, which is proportional to the amount of adjacent buses and to communication conditions. Huang & Wu (2013) analyzed observability based on the configurability control and Ahmadi et al. (2011) presented a classical approach, assuring total observability and maximizing measurement redundancy.

The Tabu Search (TS) algorithm explicitly employs its search history to escape local minima and to implement an exploratory strategy. Koutsoukis et al. (2013) employed the Recursive Tabu Search (RTS) algorithm, in which they combined the observability numerical analysis with a Greed Algorithm for initial population creation. The study of Saha Roy et al. (2012) assured that all buses had initially received a PMU and, subsequently, eliminated them by priority, according to the connectivity level of the bus. Afterwards, this work applied the Pruning method, considering that all buses must be doubly observable.

The Biogeography-Based Optimization (BBO) model is inspired in species migration from one habitat island to another. A multi-target BBO method is presented in the works of Jamuna & Swarup (2012) and Jamuna & Swarup (2011) to minimize the PMUs implementation, assuring the system observability and maximizing the redundancy measurement.

Xu et al. (2013) used the canonical CRO form and presented a simplified CRO model (SCRO) to solve OPP problem. The SCRO presented more efficient results with a simpler structure and a

shorter computational time than the canonical one. The model presented by Xu et al. (2013) was used as a reference in this investigation.

### 2.3 Chemical Reaction Optimization (CRO)

CRO, created by Lam & Li (2010), is a metaheuristic inspired by the thermodynamic laws that govern chemical reactions. These authors have applied CRO to different engineering problems in discrete and continuous domains, concluding that it presents a superior performance when compared to other optimization algorithms. A physical system is said to be unstable when it presents excess of energy. In this situation, it tends to rearrange itself through chemical reactions in order to release this excess, therefore stabilizing itself.

The CRO is a multi-agent algorithm in which the manipulated agents are a population of molecules, where each molecule owns a different problem solution. Molecule solutions suffer chemical changes that might be triggered through unimolecular or intermolecular operations, defined as elementary reactions. There are four types of elementary reactions: on-wall ineffective collision, decomposition, intermolecular ineffective collision, and synthesis. Unimolecular collisions occur when one molecule collides with an external body, such as the vessel wall, and intermolecular collisions take place when one molecule collides with another one within the system. In decomposition reactions, the molecule splits into two new molecules when colliding with an external element, and in the synthesis reactions two molecules that clash form a new one.

Any molecule exposed to an elementary reaction suffers disturbances, defined by the selected reaction change operator. The design of the CRO platform allows customization and creation of new disturbance procedures, which enables its application to different types of problems. The CRO algorithm presents 8 parameters, as described in Table 1, that defines the algorithm behavior in terms of population size, stopping criterion, definition of elementary reaction to be performed at each iteration, and the ability of the system to accept worse solutions.

When there are a small number of molecules, CRO focuses on the local search in some specific regions. Otherwise, seed is spread throughout the solution space, increasing the chance of achieving its global minimum. The compliance with synthesis and decomposition criteria presents a strong relation with the objective function behavior. If the search direction decreases, the decomposition criterion is not started. If increasing direction does not take place, the molecules will not convert their kinetic energy to worse solutions and the synthesis criterion will not be achieved.

The flowchart in Figure 1 shows the CRO algorithm, indicating those operations that intensify local search or neighborhood search, and those that diversify the search space.

**Table 1 –** CRO algorithm parameters.

| Parameter | Description | Function |
|---|---|---|
| PopSize | Population size | Initial CRO population size. |
| MAX_ITER | Stopping criterion | Defines the maximum number of CRO algorithm iterations to be executed. |
| initialKE | Initial kinetic energy | Defines the amount of initial kinetic energy for each created molecule. The kinetic energy is related to the tolerance of the system in accepting worse solutions. |
| KELossRate | Rate of kinetic energy loss | Defines the percentage of molecule kinetic energy lost at each collision suffered. |
| iniBuffer | Environment energy buffer | Sets the initial amount of energy available at the environment. |
| MoleColl | Decision variable between collision types | MoleColl is compared to a randomly generated parameter to choose between unimolecular (decomposition and on-wall ineffective collision) or intermolecular (intermolecular ineffective collision and synthesis) operations. |
| Alpha | Decision variable between unimolecular operations | Defines the decomposition criterion limit (NumHit - MinHit$> \alpha$). If the chosen molecule meets the criterion, the decomposition operation is performed; otherwise, on-wall ineffective collision takes place. |
| Beta | Decision variable between intermolecular operations | Defines the synthesis criterion limit (KE $\leq \beta$). If both chosen molecules meet this criterion, synthesis operation is performed; otherwise, intermolecular ineffective collision takes place. |

Source: Authors (2019).

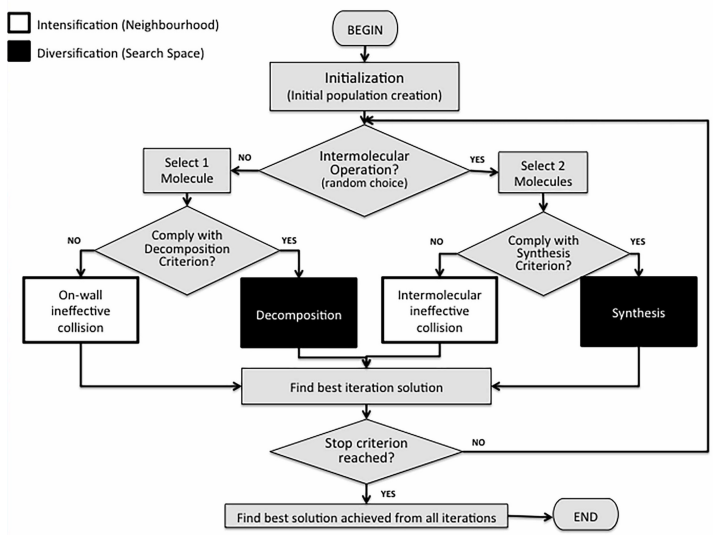


**Figure 1 –** Flowchart of the Chemical Reactions Optimization (CRO) algorithm.

Source: Authors (2019).

As follows, we describe the flowchart (Figure 1) based on Lam & Li (2010):

1. In **Initialization** CRO we create the first molecules following the pseudocode from Figure 2, whose solutions are set according to the defined strategy, such as random generation.

```
Algorithm 1 - "Molecule" class
 1: class Molecule{
 2:   Attributes:
 3:     ω, PE, KE, NumHit, MinStruct, MinPE, MinHit
 4:   Method:
 5:     Molecule() \\construtor
 6:     {
 7:       Randomly generate ω in the solition space
 8:       PE ← f (ω)
 9:       KE ← InitialKE
10:       NumHit ← 0
11:       MinStruct ← ω
12:       MinPE ← PE
13:       MinHit ← 0
14:     }
15:   OnwallIneffectiveCollision()
16:   Decomposition()
17:   IntermolecularIneffectiveCollision()
18:   Synthesis()
19: } end class
```

**Figure 2 –** Pseudocode of Molecule class.

Source: Lam & Li (2012).

2. In **Intermolecular Operation** we decide whether we apply a unimolecular or an inter-molecular collision. To do this, CRO algorithm generate a random number ***t***, in the interval of [0, 1]. If ***it*** is larger than ***MoleColl*** (a parameter of the CRO), it will result in an event of unimolecular collision. Otherwise, an inter-molecular collision will take place.

3. Next, CRO examine the **Decomposition or the Synthesis criterion** to decide which type of collision (left: **On-wall ineffective collision** or **Decomposition**; right: **Intermolecular ineffective collision** or **Synthesis**) it is. Figure 3 describes each Chemical Operation pseudocodes.

4. After that, CRO check all solutions found, identifying the best one and record it. This iteration stage repeats until the stopping criteria is reached (**Stop criterion is reached?**).

5. In the final state, CRO output is defined with the best solution found (**Find best solution achieved from all iterations**).

Xu et al. (2013) have included the observability of the system as a penalty operator in the objective function. This multi-objective function can obtain unviable solutions, which could require a higher number of software iterations to achieve optimal solutions, compromising the computational time. However, the check of the viability of the solutions became unnecessary.

At their paper, two CRO algorithm versions were applied, the canonical CRO and a simplified one (SCRO), in which a single elementary reaction is used, the on-wall ineffective collision. The

**Algorithm 2 - OnwallIneffectiveCollision()**

**Input: molecule Mω**
ω′ ← N(ω) //N(ω) - search at ω's neighbourhood
PEω′ ← f(ω)
NumHitω ← NumHitω + 1
**if** (Peω + KEω ≥ PEω′) **then**
  Generates a [KELossRate, 1]
  KEω′ ←(PEω −PEω′ +KEω)×a
  buffer ← buffer +(PEω −PEω′ +KEω)×(1−a)
  ω←ω′
  PEω ←PEω′
  :
  KEω ← KEω′
  :
  **if** (PEω < MinPEω) **then**
  :
    MinStructω ← ω
  :
    MinPEω ← PEω
  :
    MinHitω ← NumHitω
  :
  **end if**
  :
**end if**
:

**Algorithm 3 - Decomposition()**

**Input: molecule Mω**
**Creates Mω1′ and Mω2′**

Obtains ω1′ and ω2′ from ω
PEω1′ ← f(ω1′) e PEω2′ ← f(ω2′)
**if** (PEω + KEω ≥ PEω1′ + PEω2′) **then**
  Edec ← PEω +KEω −(PEω1′ +PEω2′)
  **goto** Step 13
**else**

  Generates δ1, δ2 ∈ [0, 1]
  Edec ← PEω +KEω +δ1δ2 ×buffer −(PEω1′ +PEω2′ )
  **if** Edec ≥ 0 **then**
  :
    buffer ← buffer × (1 − δ1δ2)
  :
    Generates δ3 ∈ [0, 1]
  :
    KEω1′ ←Edec ×δ3 e KEω2′ ←Edec ×(1−δ3)
    MinStructω1′ ← ω1′ e MinStructω2′ ← ω2′
  :
    MinPEω1′ ← ω1′ e MinPEω2′ ← ω2′
  :
    Destroy Mω
  :
  **else**
  :
    numHitω ← numHitω + 1
  :
    Destroy Mω1′ and Mω2′
  :
  **end if**
  :
**end if**
:

**Algorithm 4 - IntermolecularIneffectiveCollision()**

**Input: molecules Mω1 and Mω2**
ω1′ ← N(ω1) and ω2′ ← N(ω2)
PEω1′ ← N(ω1) and PEω2′ ← N(ω2)
numHitω1 ← numHitω1 + 1 and numHitω2 ← numHitω2 + 1
Einter ←(PEω1 +PEω2 +KEω1 +KEω2)− (PEω1′ +PEω2′)
**if** (Einter ≥ 0) **then**

  Generates δ4 ∈ [0, 1]
  KEω1′ ← Einter ×δ4 and KEω2′ ← Einter ×(1−δ4)
  ω1 ← N(ω1) and ω2 ← N(ω2)
  PEω1 ← PEω1′ and PEω2 ← PEω2′
  :
  KEω1 ← KEω1′ and KEω2 ← KEω2′
  :
  **if** (PEω1 < minPEω1) **then**
  :
    minStructω1 ← ω1
  :
    minPEω1 ← PEω1
  :
    minHitω1 ← numHitω1
  :
  **end if**
  :
  **if** (PEω2 < minPEω2) **then**
  :
    minStructω2 ← ω2
  :
    minPEω2 ← PEω2
  :
    minHitω2 ← numHitω2
  :
  **end if**
  :
**end if**
:

**Algorithm 5 - Synthesis()**

**Input: molecules Mω1 and Mω2**
Criates Mω′
Obtains ω′ from ω1 and ω2
**PEω′←f(ω′)**

**if** (PEω1+PEω2 + KEω1 + KEω2 ≥ PEω′) **then**

  KEω′ ←(PEω1 +PEω2 +KEω1 +KEω2)−PEω′
  minStructω′ ←ω′
  minPEω1′ ← PEω′

  Destroy Mω1 and Mω2
**else**
  :
  numHitω1 ← numHitω1 + 1 and numHitω2 ← numHitω2 + 1
  Destroy Mω′
  :
**end if**
:

**Figure 3 –** Pseudocodes for Chemical Operations of CRO Algorithm.

Source: Lam & Li (2012).

SCRO proposal can be easily understood, since the best results of the canonical form are met when the CRO parameters lead to a higher number of on-wall ineffective collisions. When the MoleColl parameter is null and Alpha is high, the choices made by the algorithm would always lead to a unimolecular reaction, followed by a high probability of on-wall ineffective collision selection.

This paper has defined that its initial population has a single molecule and, consequently, dismisses the advantages of population-based optimization, such as the diversification search in which different search space areas are visited.

### 2.4 Discussion

The OPP problem has been widely studied, being solved through different optimization techniques. The review of the literature about the observability and heuristics applied to OPP has resulted in only one paper that applied CRO to solve the OPP problem, Xu et al. (2013). It also has revealed that the numerical observability analysis is extremely more complex than the topological one, as Peng et al. (2006) stated.

As a consequence, CRO was the heuristic method selected to conduct this investigation through topological observability analysis. CRO can assume different versions according to the developed change operators or disturbance procedures. We have implemented several operators based on Lam & Li (2010), Lam & Li (2012), and Xu et al. (2013).

## 3 OPP PROBLEM MODEL

The objective function of the OPP model aims to minimize the total PMUs cost subjected to the system observability restriction. Topological observability is analyzed through matrix $\boldsymbol{H}_{nxm}$=$\boldsymbol{A.X}$, defined as the connectivity, incidence, or design matrix. This paper proposes the use of the most frequent OPP model, described by Equations (1) and (2):

$$min \sum_{i}^{n} w_i x_i, \tag{1}$$

$$subject\ to\ A.X \geq I \tag{2}$$

where $I = [111...1]^T_{Nx1}$,

$w_i = PMU\ installation\ cost\ at\ bus\ i$,

$$x_i = \begin{cases} 1, if\ there\ is\ a\ PMU\ installed\ at\ bus\ i \\ 0, otherwise \end{cases}$$

and

$$a_{ij} = \begin{cases} 1, if\ i = j \\ 1, if\ buses\ i\ and\ j\ are\ connected\ to\ each\ other \\ 0, otherwise \end{cases}$$

Let: $h(j)=a_{j1}.x_1+a_{j2}.x_2+,\ldots a_{jn}.x_n$, with $j=1,\ldots m$. If $\boldsymbol{a_{jk}}$ is zero, the product $\boldsymbol{a_{jk}}$ does not appear in $\boldsymbol{h(j)}$. If any $\boldsymbol{x_i}$ appearing in $\boldsymbol{h(j)}$ is nonzero, $\boldsymbol{h(j)}$ is observable. If all $\boldsymbol{h(j)}$ in $H$ are nonzero the system is completely observable.

As previously mentioned, (Xu et al., 2013) implemented a OPP model that includes observability ($\Phi$) as a penalty operator, with an extremely high penalty factor ($\lambda$ = 1000) in the objective function. This model was also implemented for testing purposes, being called "penalty model".

$$min \sum_{i=0}^{Nbus} S(i).C(i) + \lambda\Phi(S(i)), (\lambda \gg 1) \tag{3}$$

where $S(i) = \begin{cases} 1, \textit{if there is a PMU installed at i bar} \\ 0, \textit{otherwise} \end{cases}$,

$C(i) = \textit{cost of a PMU installed at i bar}$,

$\Phi(S(i)) = \textit{number of observability violations for S(i)}$,

where observability is verified by $\sum_{i=0}^{Nbus}\sum_{j=0}^{Nbus} A(i,j).S(i) \geq 1$, with

$$A(i,j) = \begin{cases} 1, se\ i = j \\ 1, \textit{if i and j bars are connected} \\ 0, \textit{otherwise} \end{cases}.$$

Both models use PMU costs as 1, disregarding costs influences.

### 3.1 Methods

First, the authors have specified how the population should be created. For the OPP problem, each molecule solution was implemented as a Boolean vector. The initial population solutions were created randomly, being restricted to 20 to 35% of the buses of the system with PMUs installed. This limitation follows Baldwin et al. (1993), which states that the optimal PMU number lies between 20 and 30% of the total number of buses in the electrical system. However, the upper limit was set to 35% in order to decrease the execution time required to search for OPP solutions that can cope with the constraints (2).

CRO elementary reactions were developed as per the pseudocodes presented by Lam & Li (2012). The implementation of the disturbance procedures for each CRO elementary reaction define how the intensification and diversification will take place. Table 2 describes the routines created and applied to achieve the presented results.

Random 2XChange routine was employed in the on-wall and intermolecular ineffective collision reactions. In the decomposition reactions HalfRandom Change was employed. In the synthesis reactions, uses OnePositionXChange routine. For each reaction, we inserted a loop of 10 attempts to search for a new solution that meets the problem restriction (2), where the respective

**Table 2 –** Disturbance routines applied.

| Routine | Description | Source |
|---|---|---|
| 2XChange | Select a random bit from the molecule solution (w) and change its value for a smaller position bit value. | Lam & Li (2012) |
| Random 2XChange | Perform a random number (between 0 and 50) of bit pair exchanges in the molecule solution (w). The positions of the bits that are changed are randomly selected. | Authors |
| OneResource Change | Randomly selects two positions from the molecule solutions array (w). Looks for the first bit that is set after the first position and for the first not-set bit after the second position. Both bits are exchanged for the output solution. | Lam & Li (2011) |
| OneResourceRandomChange | Select a random bit from the molecule solution (w) and change its value | Lam & Li (2011) |
| HalfRandom Change | Creates two new binary solutions (w1 and w2) similar to the original molecule (w). Changes the even bits values for molecule w1 and changes the odd bits values for molecule w2. | Lam & Li (2011) |
| OnePosition XChange | Selects a random position of the array as a cut-off point to be applied at the input molecules (w1 and w2). Creates a new molecule that combines the solutions from w1 and w2. Its solution incorporates all the bits before the cut-off point from w1, discarding the rest of the array, and concates them with all the bits after the cut-off point of w2. | Lam & Li (2011) |
| Exhaustive Change | Runs every position from the molecule solution (w), changing its value and testing its results. If a best result is met, it becomes the new solution (w'). It continues the search at the new solution (w') until the last array position is verified. | Authors |

Source: Authors.

disturbance procedure was run. During this loop, if any feasible solution was found, this solution was assumed as the output one and the disturbance loop was broken. On the other hand, if no feasible solution was found after 10 attempts, the initial solutions were kept as output.

In order to parameterize the CRO algorithm, the range of values for the 8 algorithm parameters, described at Table 1, was empirically set. Several simulations were run combining them, reaching more than 3,500 ***combination tests***. If a set of ***combination tests*** considers, for example, 2 values of **Iterations** x 2 values of **Alpha** x 2 values of **Beta** x 3 values of **KELoss Rate** x 1 values of **Buffer**, there would be a total of 24 combinations. The values of each parameter are described in the Table 3. In the simulations, we have used instances of 14, 30, 57 and 188 buses of IEEE power systems test cases library and carried out 15 times each instance.

Each value range was defined by observing the behavior of the variables, the correlation of errors, and the percentage of occurrence of the reactions, as exemplified in Figure 4.

The intervals of the beta parameter were extended in order to increase the occurrence of both the ineffective intermolecular collision and the synthesis reactions. In the same way, lower Alpha values were applied to promote the occurrence of the decomposition reactions; higher KELoss-

**Table 3 –** CRO Parameters values for parameterization tests.

| Parameter | Values |
|---|---|
| PopSize | 1, 10, 100 |
| MAX_ITER | 100, 500, 1000 *(number of nodes from the test files) |
| iniBuffer | 3000, 5000, 10000 |
| initialKE | Molecule initial PE, set as objective function value |
| KELossRate | 0,3, 0.5, 0.8 |
| MoleColl | 0.2, 0,3, 0.5, 0.8, 0.9 |
| Alpha | 10, 100, 200*( number of nodes from the test files) |
| Beta | 200, 500, 800*( number of nodes from the test files) |

Source: Author (2020).

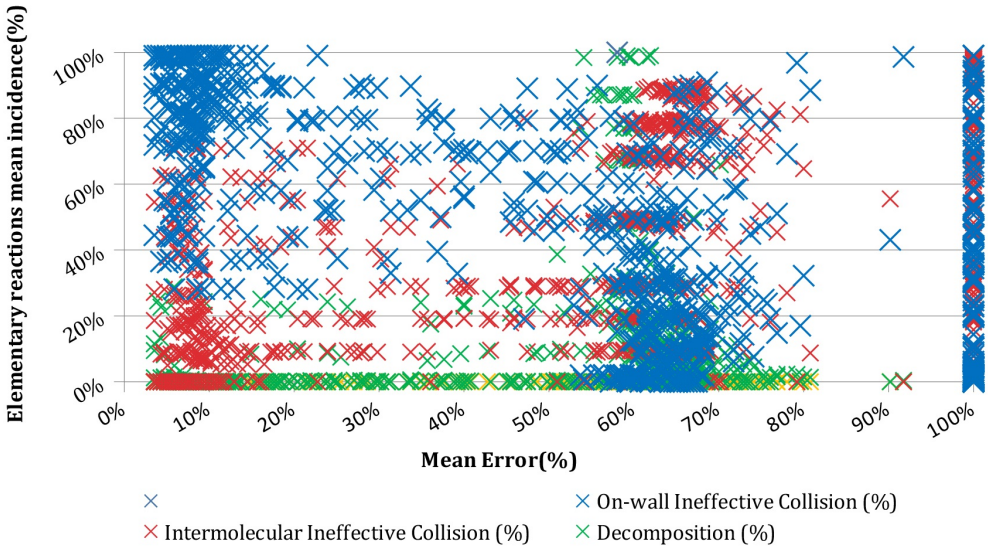


**Figure 4 –** Percentage of occurrence of the elementary reactions versus mean error.

Source: Authors (2019).

Rate values were used to enable worse solutions in different search areas; and higher values of the Buffer parameter were tested to enable operations that consume energy from the environment, increasing the feasibility of changes that diversify the search space.

For the tests performed with the model of Xu et al. (2013), named "penalty model", initial solutions were set as randomly generated Boolean vectors, without any restriction; and both operations of ineffective collision reactions, OnWall and Intermollecular, were implemented through a single random bit change, OneResourceRandomChange routine from Table 2. Decomposition applies HalfRandomChange and Synthesis uses OnePositionXChange routine from Table 2. At this implemented model, the observability constraint is inserted in the objective function, as per Equation (3), so it was not necessary to check whether each solution change met the observ-

ability restriction (2). The parametrization followed the same methodology adopted in the model proposed in Equations (1) and (2).

The algorithms were developed in Java using the NetBeans 8.2 software. The tests were run in an Intel Core i7 3.6GHz 16GB RAM computer with Windows 10 operating system. Each parameter combination was executed 15 times. The best solution found, the mean error, and the average runtime were recorded.

### 3.2 Computational Results

The best results found through the different parameter combinations, for both the Penalty model and the proposed model in Equations (1) and (2), are shown in Table 4.

"BS" cells indicate the best solution found for each PMU number, for each instance of the IEEE test systems. In Table 4 was used a metric that it represents the average percentage error reached by algorithm considering the best values in Xu et al. (2013). This metric shows the robustness of the algorithm provided in the all executions. "AE(%) Penalty" represents the average percentage error reached by the penalty model, which is the model proposed by Xu et al. (2013) that applies observability as a penalty at the objective function (equations (3)), as described in start of section 3. "AE(%) Model (1)(2)" represents the average percentage error found for the model proposed by equations (1) and (2).

The first results were reached with the MoleColl parameter null, which prevents the occurrence of intermolecular reactions. Since only unimolecular reactions were performed, the synthesis criterion was never checked, therefore, lower Beta values were accepted in these cases. The best results were found for the highest KELossRate values, which indicate that the algorithm accepts worse solutions.

The proposed model according to Equations (1) and (2) reaches optimal solutions with lower percentage errors than the penalty model. However, the best results were still found with the higher incidence of local search reactions (on-wall and intermolecular ineffective collisions) and the lower incidence of the decomposition and synthesis reactions. For comparison purposes, the average runtime achieved for each model in the IEEE 118 bus system was compared.

The lowest computational time (376 ms) in this paper presented was obtained when running the penalty model, according to Equations (3) and (4) with Intel Core i7 3.6GHz 16GB RAM. Xu et al. (2013) have reached the optimal solution in 3.1s using the canonical CRO, and 1.1s with SCRO with an Intel Core Duo 2.66 Hz CPU and 2 GB RAM. Both computers run one stream of instructions rather than multiple parallel streams per core. Thus, using the metric "CPU Single Thread Rating" (PassMark, 2020) for comparison sees Table 5.

With the proposed model, as shown in Equations (1) and (2), this work has reached a runtime of 991ms in the IEEE 118 bus system. The present research has focused on the quality of the solution instead of the algorithm computational time. However, in this bus system was considered irrelevant.

**Table 4 –** CRO best results for IEEE power system test cases.

| CRO Parameters | | | | | | | IEEE 14 Buses | | IEEE 30 Buses | | IEEE 57 Buses | | IEEE 118 Buses | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | BS: 4 PMUs | | BS: 10 PMUs | | BS: 17 PMUs | | BS: 32 PMUs | |
| Population Size | MAX_ITER | Alpha | Beta | KELoss Rate | MoleColl | Buffer | AE (%) Penalty | AE (%) Model (1)(2) | AE (%) Penalty | AE (%) Model (1)(2) | AE (%) Penalty | AE (%) Model (1)(2) | AE (%) Penalty | AE (%) Model (1)(2) |
| 1 | 500 | 200 | 20 | 0.9 | 0.0 | 1000 | 0.00 | 0.00 | 20.00 | 0.00 | 9.02 | 0.00 | 9.17 | 0.63 |
| 1 | 700 | 150 | 50 | 0.8 | 0.0 | 0 | 3.33 | 0.00 | 30.00 | 0.00 | 4.31 | 0.00 | 9.79 | 0.42 |
| 10 | 700 | 150 | 2 | 0.7 | 0.0 | 1000 | 3.33 | 0.00 | 15.33 | 0.00 | 8.24 | 0.00 | 11.25 | 1.04 |
| 10 | 700 | 150 | 500 | 0.8 | 0.0 | 1000 | 0.00 | 0.00 | 25.33 | 0.00 | 6.67 | 0.00 | 9.38 | 2.29 |
| 10 | 700 | 500 | 2 | 0.9 | 0.0 | 0 | 6.67 | 0.00 | 22.67 | 0.00 | 7.84 | 0.00 | 9.17 | 2.29 |
| 10 | 700 | 500 | 50 | 0.9 | 0.0 | 1000 | 10.00 | 0.00 | 23.33 | 0.00 | 6.67 | 0.00 | 9.58 | 1.67 |
| 1 | 100 | 100 | 500 | 0.8 | 0.2 | 10000 | 21.67 | 0.00 | 7.33 | 0.00 | 13.33 | 0.00 | 14.17 | 4.38 |
| 1 | 100 | 200 | 800 | 0.8 | 0.3 | 5000 | 18.33 | 0.00 | 9.33 | 0.00 | 11.76 | 1.18 | 14.58 | 3.33 |
| 1 | 500 | 200 | 500 | 0.8 | 0.3 | 3000 | 26.67 | 1.67 | 18.67 | 0.00 | 14.90 | 0.78 | 19.58 | 1.04 |
| 10 | 500 | 100 | 200 | 0.8 | 0.5 | 5000 | 18.33 | 26.67 | 7.33 | 0.67 | 12.94 | 1.18 | 17.92 | 1.46 |
| 100 | 1000 | 100 | 800 | 0.8 | 0.2 | 10000 | 15.00 | 36.67 | 7.33 | 0.67 | 10.20 | 2.35 | 15.00 | 1.04 |
| 100 | 1000 | 200 | 200 | 0.8 | 0.5 | 5000 | 13.33 | 1.67 | 6.00 | 0.00 | 12.55 | 0.78 | 15.83 | 0.83 |

Source: Authors (2019).

**Table 5 –** Comparison between computers.

| | **Our paper** | **Xu et al. (2013)** |
|---|---|---|
| CPU | Intel Core i7 3.6GHz | Intel Core Duo 2.66 Hz |
| CPU Single Thread Rating | 2904 | 743 |
| Time IEEE 118 bus system (seconds) - algorithm | 0.376 s - CRO | 3.1s - CRO 1.1s - SCRO |
| **Time IEEE 118 bus system (seconds) normalized** | **1.46 s - CRO** | **3.1s - CRO 1.1s - SCRO** |

Source: Authors (2020).

### 3.3 Discussion

The developed software for the proposed model, set by Equations (1) and (2), reached best known solutions for the IEEE power system test cases (14, 30, 57 and 118 buses) with better error rates and in a lower computational time than the software using the penalty model (0.376s against 1.46s for IEEE 118 buses test case). However, running time was considered irrelevant due to the software purposes.

The best results still presented a higher frequency of local search reactions than global search ones, especially for on-wall ineffective collision reactions. This corroborates Xu et al. (2013) regarding the SCRO model efficiency for the OPP problem.

## 4 SET COVERING PROBLEM

The Set Covering Problem goal is to find the subset combination with the lowest cost that is able to cover a given search space; considering that each subset covers a given point set in space.

We handled OPP problem as a Set Covering problem, once a set of buses is chosen to guarantee measured or calculated data from the electrical network. The developed CRO algorithm for OPP problems was tested in SCPs in order to check its behavior in large scale problems. (Beasley, 1990) provides an OR-Library for Set Covering Problems with large size instances to evaluate the large-scale problems.

The work of Yu et al. (2014) has applied CRO to OR-Library, therefore it was used as the reference work for the tests with SCP instances. Table 6 shows Beasley OR-Library instances details.

**Table 6 –** Beasley OR-Library test groups.

| Test Group Number | Number of instances by group | Problem Scale | Cost Variation | Density | Optimal solution is known? (Yes/No) |
|---|---|---|---|---|---|
| 4 | 10 | 200 x 1000 | 1-100 | 2% | Yes |
| 5 | 10 | 200 x 2000 | 1-100 | 2% | Yes |
| 6 | 5 | 200 x 1000 | 1-100 | 5% | Yes |
| A | 5 | 300 x 3000 | 1-100 | 2% | Yes |
| B | 5 | 300 x 3000 | 1-100 | 5% | Yes |
| C | 5 | 400 x 4000 | 1-100 | 2% | Yes |
| D | 5 | 400 x 4000 | 1-100 | 5% | Yes |
| NRE | 5 | 500 x 5000 | 1-100 | 10% | No |
| NRF | 5 | 500 x 5000 | 1-100 | 20% | No |
| NRG | 5 | 1000 x 10000 | 1-100 | 2% | No |
| NRH | 5 | 1000 x 10000 | 1-100 | 5% | No |
| E | 5 | 50 x 500 | - | 2% | Yes |
| CLR.10 | 1 | 511 x 210 | - | 2% | No |
| CLR.11 | 1 | 1023 x 330 | - | 5% | No |
| CLR.12 | 1 | 2047 x 495 | - | 2% | No |
| CLR.13 | 1 | 4095 x 715 | - | 5% | No |
| CYC.6 | 1 | 240 x 192 | - | 2% | Yes |
| CYC.7 | 1 | 672 x 448 | - | 5% | No |
| CYC.8 | 1 | 1792 x 1024 | - | 10% | No |
| CYC.9 | 1 | 4608 x 2304 | - | 20% | No |
| CYC.10 | 1 | 11520 x 5120 | - | 2% | No |
| CYC.11 | 1 | 28160 x 11264 | - | 5% | No |

Source: Adapted from Yu et al. (2014).

### 4.1 Set Covering Model

The Set Covering model presented by Yu et al. (2014) was implemented at this investigation for test purposes, following Equations (4) and (5), where $S_j$ is a subset that covers a certain space and $a_{ij}$ displays the search space points covered by each subset. However, Yu et al. (2014) uses Real vectors instead of Boolean vectors to represent the SCP solution.

$$min \sum_{j}^{n} c_j x_j, \tag{4}$$

$$subject\ to\ a_{ij} x_j \geq I \tag{5}$$

where $x_j = \begin{cases} 1, if\ S_j is\ a\ solution\ subset \\ 0,\ otherwise \end{cases}$,

$c j = S_j\ subset\ cost$,

$a_{ij} = \begin{cases} 1, if i \in S_j, or\ if\ i\ element\ is\ covered\ by\ j\ subset \\ 0, otherwise. \end{cases}, i \leq m$ .

$I = [111...1]^T_{Nx1}$.

## 4.2 Methods

### 4.2.1 Initial Tests

The initial solutions of the population were set as Boolean vectors in which the bits indicate which subset is part of the solution. Initially, the vectors were created randomly, without any restriction. After performing tests with the Beasley scp41 test file, similarly to the IEEE tests, the authors have inserted a restriction, setting only 20 to 35% of the subsets as part of the initial solution. Likewise, if a created solution is already present at the population, a new solution is generated.

After running the ***scp41, scp51, scp61*** file tests, other Beasley SCP types were tested as ***scpe1, scpe2***, and groups of ***a, b, c, d, e , and nre***. For some types, the computational time spent to find a solution that meets the coverage constraint (5) and the limit set to 20 to 35% of the selected subsets were considered too high. For such files, the restriction of the selected subsets was removed.

The change operators, or disturbance procedures, for the elementary reactions were the same used for the IEEE test files. Due to the struggle to find new solutions that meet the coverage constraint (5) after the disturbance of the elementary reactions, if the constraint is not reached, the disturbance is run up to 100 times. After that, if a solution in accordance to the constraint is still not found, the initial solution was taken as output. Otherwise, if at any time feasible solutions were found by the disturbance procedures, their results were taken as output and the loop was broken before reaching the limit.

In order to improve the initial and final solutions, a local search that changes a random number of bit pairs, 2XChange routine at Table 2, was run 10 times after each molecule creation. After the CRO result an exhaustive search, Exhaustive Change routine at Table 2, where after each bit change the results are checked for improvement, is performed 10 times. After each execution of local searches, if a new solution that fits the constraint with better results is found, it is assumed as the new solution.

There are several instances in Beasley library, leading to a high computational time for the parametrization of each one. Therefore, the parametrization was restricted to the scp41 file, applying its parameters to all the other file types.

The same methodology used in the IEEE test parametrization was employed in SCP, including the same tested parameter combinations. The best results of the CRO parameters scp41 are shown in Table 7 and were applied to all the Beasley library test files.

For each Beasley file, CRO was run 15 times with Table 7 parameters, recording the best solution, mean solution value, mean error, and mean computational time found. Tests were performed for 31 of the 80 instances available in the Beasley library. The results are presented at Table 8.

**Table 7 –** Defined CRO parameters for SCP.

| Parameter | Values |
|---|---|
| PopSize | 10 |
| MAX_ITER | 15*(Number of columns of the Set Covering problem) |
| iniBuffer | 10000 |
| initialKE | 50000 |
| KELossRate | 0.3 |
| MoleColl | 0.3 |
| Alpha | 0.5*(Number of columns of the Set Covering problem) |
| Beta | 1*(Number of columns of the Set Covering problem) |

Source: Authors (2019).

**Table 8 –** Results for Beasley library test files.

| Instance | Mean Computational Time (s) | Lines (Search Space elements) | Columns (available subsets) | Best Known Solution Yu et al. (2014) | Best Found Solution | GAP (%) | Mean Solution Value | Mean Error (%) |
|---|---|---|---|---|---|---|---|---|
| 41 | 167 | 200 | 1000 | 429 | 432 | 0.70% | 449.60 | 4.80% |
| 42 | 168 | 200 | 1000 | 512 | 525 | 2.54% | 555.40 | 8.48% |
| 43 | 160 | 200 | 1000 | 516 | 531 | 2.91% | 546.53 | 5.92% |
| 44 | 175 | 200 | 1000 | 494 | 530 | 7.29% | 543.93 | 10.11% |
| 45 | 176 | 200 | 1000 | 512 | 522 | 1.95% | 548.40 | 7.11% |
| 46 | 169 | 200 | 1000 | 560 | 568 | 1.43% | 592.33 | 5.77% |
| 410 | 161 | 200 | 1000 | 514 | 548 | 6.61% | 563.33 | 9.60% |
| 51 | 1077 | 200 | 2000 | 253 | 271 | 7.11% | 278.27 | 9.99% |
| 55 | 1085 | 200 | 2000 | 211 | 221 | 4.74% | 225.53 | 6.89% |
| 510 | 1054 | 200 | 2000 | 265 | 270 | 1.89% | 276.07 | 4.18% |
| 61 | 299 | 200 | 1000 | 138 | 144 | 4.35% | 150.47 | 9.03% |
| 65 | 286 | 200 | 1000 | 161 | 171 | 6.21% | 176.93 | 9.90% |
| a4 | 4296 | 300 | 3000 | 234 | 246 | 5.13% | 254.40 | 8.72% |
| a5 | 4268 | 300 | 3000 | 236 | 240 | 1.69% | 247.20 | 4.75% |
| b1 | 5466 | 300 | 3000 | 69 | 77 | 11.59% | 81.73 | 18.45% |
| b5 | 5444 | 300 | 3000 | 72 | 75 | 4.17% | 80.93 | 12.41% |
| c1 | 10108 | 400 | 4000 | 227 | 232 | 2.20% | 242.73 | 6.93% |
| c5 | 10475 | 400 | 4000 | 215 | 224 | 4.19% | 233.20 | 8.47% |
| clr10 | 29 | 511 | 210 | 25 | 30 | 20.00% | 31.73 | 26.93% |
| clr11 | 132 | 1023 | 330 | 23 | 31 | 34.78% | 32.33 | 40.58% |
| d1 | 13158 | 400 | 4000 | 60 | 65 | 8.33% | 69.73 | 16.22% |
| d5 | 12564 | 400 | 4000 | 61 | 66 | 8.20% | 71.73 | 17.60% |
| e1 | 34 | 50 | 500 | 5 | 6 | 20.00% | 6.87 | 37.33% |
| e2 | 35 | 50 | 500 | 5 | 6 | 20.00% | 6.87 | 37.33% |
| e3 | 36 | 50 | 500 | 5 | 6 | 20.00% | 6.53 | 30.67% |
| e5 | 36 | 50 | 500 | 5 | 6 | 20.00% | 6.80 | 36.00% |
| nre4 | 31434 | 500 | 5000 | 28 | 31 | 10.71% | 33.53 | 19.76% |
| nre5 | 31369 | 500 | 5000 | 28 | 31 | 10.71% | 32.93 | 17.62% |
| nrf1 | 45501 | 500 | 5000 | 14 | 16 | 14.29% | 17.87 | 27.62% |

Source: Authors (2019).

The GAP values varied according to the test file type and the processing time grew exponentially following the problem scale. For high-scale Beasley instances, the defined stopping criterion (MAX_ITER), directly dependent on the number of columns of the file, made the software run at a low pace. The effect is easily noted for NRG and NRH files, which have 10,000 columns each, reaching 150,000 iterations for the stopping criterion in each CRO execution.

The reference study, Yu et al. (2014), does not show the time required by the CRO algorithm to reach the displayed results. Therefore, it cannot be used as a base to check the behavior. When comparing computational times to Sundar & Singh (2012), that apply hybrid heuristic to solve SCP problem, we notice that we need to improve our software efficiency. For instance 41 they achieved the best known solution in 5.3s against 167s of our CRO and for nrf1 test file they were able to reach the best known solution at 330.41s while we got a GAP of 14.29% at 45501s.

In order to identify how to improve our software, we performed some tests to evaluate the impact of disturbance repeat loops and stopping criterion values in its behavior.

### 4.2.2 Disturbance Repeat Limits

First, the authors have decided to change the disturbance repeat rates for all chemical reaction. Initially, if a solution could not be found in 100 disturbance attempts, the solutions of the initial molecules were maintained.

eIf after a disturbance the solution of the molecule does not fit into the model restriction, it was submitted again to the disturbance procedure. This process was repeated until a defined repeat limit value was reached. The authors have tested 3 disturbances repeat limit values (10, 20, and 50 times) up to 100 times, with CRO parameters according to Table 7 - Defined CRO parameters for SCP. The results are shown in Table 9.

**Table 9 –** Tests results for disturbance repeat limit values.

| Instance | Disturbance Limit | Computational time (s) | Best Known Solution Yu et al. (2014) | Best Solution Found | Error (%) | Mean Solution Values | Mean Error (%) |
|---|---|---|---|---|---|---|---|
| 42 | 10 | 111 | 512 | 2755 | 438.09 | 569.80 | 11.29 |
| 42 | 20 | 143 | 512 | 1587 | 209.96 | 559.60 | 9.30 |
| 42 | 50 | 209 | 512 | 530 | 3.52 | 555.40 | 8.48 |
| 65 | 10 | 116 | 161 | 178 | 10.56 | 819.50 | 223.91 |
| 65 | 20 | 147 | 161 | 171 | 6.21 | 982.70 | 288.42 |
| 65 | 50 | 236 | 161 | 173 | 7.45 | 184.10 | 14.35 |
| e1 | 10 | 82 | 5 | 7 | 40.00 | 8269.30 | 3746.19 |
| e1 | 20 | 88 | 5 | 7 | 40.00 | 6990.60 | 3151.44 |
| e1 | 50 | 103 | 5 | 6 | 20.00 | 5471.30 | 2444.79 |
| clr11 | 10 | 128 | 23 | 31 | 34.78 | 8925.20 | 14531.48 |
| clr11 | 20 | 160 | 23 | 31 | 34.78 | 8750.80 | 14245.57 |
| clr11 | 50 | 246 | 23 | 31 | 34.78 | 6667.80 | 10830.82 |
| clr11 | 50 | 246 | 23 | 31 | 34.78 | 6667.80 | 10830.82 |

Source: Authors (2019).

When analyzing the parametrization backgrounds a direct relation is noted: the higher the repeat limit value the better is the solution found and, as a result, the lower is the error compared to the best-known solution. In contrast, the worst is the computational time spent to reach a model solution.

### 4.2.3 Stopping Criterion Tests

In order to avoid high running times, concurrently to disturbance repeat limits tests, MAX_ITER values have also been tested for each type of SCP. For these tests the disturbance repeat limit was set to 50 and the applied CRO parameters were established according to Table 7. The results can be seen at Table 10. In most cases, better results were reached for higher MAX_ITER values, when CRO is run more times.

**Table 10 –** Tests results for MAX_ITER values.

| Instance | Number of nodes | CRO Iteration Number | Computational time (s) | Best Known Solution Yu et al. (2014) | Best Solution Found | Error (%) | Mean Solution Values | Mean Error (%) |
|---|---|---|---|---|---|---|---|---|
| 51 | 1000 | 10000 | 272 | 253 | 608 | 140.32 | 2414.0 | 854.15 |
| 51 | 1000 | 15000 | 406 | 253 | 274 | 8.30 | 1618.4 | 539.68 |
| 51 | 1000 | 20000 | 516 | 253 | 277 | 9.49 | 819.5 | 223.91 |
| 51 | 1000 | 25000 | 649 | 253 | 278 | 9.88 | 982.7 | 288.42 |
| 65 | 2000 | 10000 | 159 | 161 | 174 | 8.07 | 184.1 | 14.35 |
| 65 | 2000 | 15000 | 232 | 161 | 175 | 8.70 | 180.0 | 11.80 |
| 65 | 2000 | 20000 | 319 | 161 | 175 | 8.70 | 180.0 | 11.80 |
| 65 | 2000 | 25000 | 402 | 161 | 166 | 3.11 | 178.8 | 11.06 |
| a4 | 1000 | 10000 | 541 | 234 | 2675 | 1043.16 | 5855.8 | 2402.48 |
| a4 | 1000 | 15000 | 835 | 234 | 2226 | 851.28 | 5854.0 | 2401.71 |
| a4 | 1000 | 20000 | 1133 | 234 | 1963 | 738.89 | 4644.1 | 1884.66 |
| a4 | 1000 | 25000 | 1379 | 234 | 241 | 2.99 | 1874.6 | 701.11 |
| e1 | 4000 | 10000 | 67 | 5 | 6 | 20.00 | 7.4 | 48.00 |
| e1 | 4000 | 15000 | 102 | 5 | 6 | 20.00 | 6.9 | 38.00 |
| e1 | 4000 | 20000 | 137 | 5 | 6 | 20.00 | 6.8 | 36.00 |
| e1 | 4000 | 25000 | 171 | 5 | 6 | 20.00 | 6.6 | 32.00 |
| clr11 | 500 | 10000 | 162 | 23 | 31 | 34.78 | 31.9 | 38.70 |
| clr11 | 500 | 15000 | 245 | 23 | 31 | 34.78 | 32.3 | 40.43 |
| clr11 | 500 | 20000 | 327 | 23 | 31 | 34.78 | 32.0 | 39.13 |
| clr11 | 500 | 25000 | 410 | 23 | 31 | 34.78 | 31.6 | 37.39 |

Source: Authors (2019).

## 4.3 Discussion

Despite reaching GAPs lower than 10% for some Beasley OR-Library tests, developed software needs to be improved to SCP problems. The results are likely to show an inverse relation between GAP values and higher CRO stopping criterion values, resulting in impractical computational times. Sundar & Singh (2012) achieved the best known solutions at better computation times.

Both Yu et al. (2014) and Sundar & Singh (2012) use a solution representation based in Real numbers vector, indicating the numbers of the selected sub-sets. At this paper the solution is a

Boolean vector where each position corresponds to a sub-set that will cover part of the search space. If the vector position is set to 1, the sub-set is selected, and if 0, otherwise. This design leads to high processing times, specially at constraints verification where time complexity reaches $O(n^2)$, with n as the problem scale (nodes number).

More tests, as a new solution representation, are required to evaluate CRO optimization method efficiency to solve SCP problem. By the current results, this heuristic would not fit the problem.

## 5 CONCLUSION AND FINAL REMARKS

The results obtained for the OPP problem with the CRO algorithm applied to the IEEE test systems of 14, 30, 57, and 118 bus achieved the best known solution values, as presented in this paper. The results show a higher occurrence of CRO elementary reactions that perform intensification, or local-search, specially "on-wall ineffective collision". This is adherent to a version of CRO algorithm (SCRO) from Xu et al (2013) that sustains the use of this single reaction. The computational time is considered acceptable and irrelevant in order of seconds. Computational results were also considered satisfactory in non-unitary populations, unlike than that was proposed by Xu et al (2013).

The application of the developed CRO algorithm to SCP problem aimed to check its performance in large scale problems. However, the results suggest that the CRO is not the best heuristic to be used for this problem optimization with the number of iterations described in Table 8.

On the other hand, CRO performance to the SCP problems was reasonable for some large size instances (1000 to 2000 columns). We suggest running more tests using CRO to solve SCP problem, particularly regarding solution structures, using Real vectors instead of Boolean ones to reduce computational times.

The probability of PMUs placement in an electrical system of the magnitude of Beasley OR-Library for SCP problems, where system bus number would be larger than 2000 buses, is relatively low. For this reason, CRO is still regarded as a great optimization method for OPP because it determines a minimum number of PMUs placement required in a power system.

### Acknowledgements

The authors would like to thank the reviewer for the constructive feedback. The authors also acknowledge the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES).

### References

[1] AHMADI A, ALINEJAD-BEROMI Y & MORADI M. 2011. Optimal PMU placement for power system observability using binary particle swarm optimization and considering measurement redundancy. *Expert Systems with Applications*, **38**(6): 7263—7269. Available at: https://doi.org/10.1016/j.eswa.2010.12.025

[2] BALDWIN TL, MILI L, BOISEN MB & ADAPA R. 1993. Power System Observability with Minimal Phasor Measurement Placement. *IEEE Transactions on Power Systems*, **8**(2): 707—715. Available at: https://doi.org/10.1109/59.260810

[3] BEASLEY JE. 1990. OR-Library for Set Covering test data sets. Available at: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/scpinfo.html. Accessed at: 03/14/2018.

[4] BEDEKAR PP, BHIDE SR & KALE VS. 2011. Optimum PMU placement considering one Line/ One PMU outage and maximum redundancy using genetic algorithm. *The 8th Electrical Engineering/ Electronics, Computer, Telecommunications and Information Technology (ECTI) Association of Thailand - Conference 2011*, p. 688–691. Available at: https://doi.org/10.1109/ECTICON.2011.5947933

[5] CAVALHEIRO EM, VERGÍLIO AH & LYRA C. 2018. Optimal configuration of power distribution networks with variable renewable energy resources. *Computers & Operations Research*, **96**: 272–280. Available at: https://doi.org/10.1016/j.cor.2017.09.021.

[6] BRUENI DJ & HEATH LS. 2005. The PMU Placement Problem. *SIAM Journal on Discrete Mathematics*, **19**(3): 943–946.

[7] HAJIAN M, RANJBAR AM, AMRAEE T & MOZAFARI B. 2011. Optimal placement of PMUs to maintain network observability using a modified BPSO algorithm. *International Journal of Electrical Power and Energy Systems*, **33**(1): 28–34. Available at: https://doi.org/10.1016/j.ijepes.2010.08.007

[8] HUANG J & WU NE. 2013. Fault-tolerant placement of phasor measurement units based on control reconfigurability. *Control Engineering Practice*, **21**(1): 1–11. Available at: https://doi.org/10.1016/j.conengprac.2012.09.001

[9] HUI-LING Z, YUAN-XIU D, XIAO-PAN Z, HUAN Q & CHENG-XUN H. 2013. Hybrid of MST and Genetic Algorithm on Minimizing PMU Placement. *2013 Third International Conference on Intelligent System Design and Engineering Applications*, 820–823. Available at: https://doi.org/10.1109/ISDEA.2012.195

[10] IEEE. Power Systems Test Cases. Available at: https://www2.ee.washington.edu/research/pstca/. Accessed at: 11/29/2017.

[11] JAMUNA K & SWARUP KS. 2011. Power system observability using biogeography based optimization. *International Conference on Sustainable Energy and Intelligent Systems (SEISCON 2011)*, p. 384–389. Available at: https://doi.org/10.1049/cp.2011.0394

[12] JAMUNA K & SWARUP KS. 2012. Multi-objective biogeography based optimization for optimal PMU placement. *Applied Soft Computing*, **12**(5): 1503–1510. Available at: https://doi.org/10.1016/j.asoc.2011.12.020

[13] KOUTSOUKIS NC, MANOUSAKIS NM, GEORGILAKIS PS & KORRES GN. 2013. Numerical observability method for optimal phasor measurement units placement using recursive tabu search method. *IET Generation, Transmission and Distribution*, **7**(4): 347–356. Available at: https://doi.org/10.1049/iet-gtd.2012.0377

[14] LAM AYS & LI VOK. 2010. Chemical-reaction-inspired metaheuristic for optimization. *IEEE Transactions on Evolutionary Computation*, **14**(3): 381–399. Available at: https://doi.org/10.1109/TEVC.2009.2033580

[15] LAM AYS & LI VOK. 2012. Chemical Reaction Optimization: A tutorial. *Memetic Computing*, **4**(1): 3–17. Available at: https://doi.org/10.1007/s12293-012-0075-1

[16] LIAO C-S., HSIEH T-J., GUO X-C., LIU J-H. & CHU C-C. 2015. Hybrid search for the optimal PMU placement problem on a power grid. *European Journal of Operational Research*, **243**(3): 985–994. Available at: https://doi.org/10.1016/j.ejor.2014.12.047.

[17] MOHAMMADI-IVATLOO B. 2009. Optimal Placement of PMUs for Power System Observability Using Topology Based Formulated Algorithms. *J Appl Sci*, **9**(13): 2463–2468. Available at: http://www.docsdrive.com/pdfs/ansinet/jas/2009/2463-2468.pdf

[18] MOHAMMADI MB, HOOSHMAND RA & FESHARAKI FH. 2016. A new approach for optimal placement of PMUs and their required communication infrastructure in order to minimize the cost of the WAMS. *IEEE Transactions on Smart Grid*, **7**(1): 84–93. Available at: https://doi.org/10.1109/TSG.2015.2404855

[19] NAZARI-HERIS M, MOHAMMADI-IVATLOO B. 2015. Application of heuristic algorithms to optimal PMU placement in electric power systems: An updated review. *Renewable and Sustainable Energy Reviews*, **50**: 214–228.

[20] NIST. 2012. *NIST PAP2 guidelines for assessing wireless standards for smart grid application*.

[21] PASSMARK. 2020. CPU Benchmarks. PassMark Software Pty Ltda. Available at: https://www.cpubenchmark.net/compare/. Accessed at: 05/01/2020.

[22] PENG J ET AL. 2006 Optimal PMU placement for full network observability using Tabu search algorithm. *International Journal of Electrical Power and Energy Systems*, **28**(4): 223–231.

[23] PEREIRA ED, COELHO AS, LONGARAY AA, MACHADO CM & MUNHOZ PR. 2018. Metaheuristic Analysis Applied to the Berth Allocation Problem: Case Study in a Port Container Terminal. *Pesquisa Operacional*, **38**(2): 247–272. Available at: https://dx.doi.org/10.1590/0101-7438.2018.038.02.0247.

[24] SAHA ROY BK, SINHA AK & PRADHAN AK. 2012. An optimal PMU placement technique for power system observability. *International Journal of Electrical Power and Energy Systems*, **42**(1): 71–77. Available at: https://doi.org/10.1016/j.ijepes.2012.03.011

[25] SILVA DC & MESTRIA M. 2018. Chemical Reaction Optimization Metaheuristic for Locating Service Stations through the Capacitated P-Median Problem. *Pesquisa Operacional*, **38**(3): 441–478. Available at: https://doi.org/10.1590/0101-7438.2018.038.03.0441.

[26] SU C & CHEN Z. 2010. Optimal Placement of Phasor Measurement Units with New Considerations. *2010 Asia-Pacific Power and Energy Engineering Conference*, p. 1–4. Available at: https://doi.org/10.1109/APPEEC.2010.5448192

[27] SUNDAR S & SINGH A. 2012. A hybrid heuristic for the set covering problem. *Oper Res Int J* **12**: 345–365. doi: 10.1007/s12351-010-0086-y

[28] XU J, WEN MHF, LI VOK & LEUNG K-C. 2013. Optimal PMU Placement for Wide-Area Monitoring Using Chemical Reaction Optimization. *2013 IEEE PES Innovative Smart Grid Technologies Conference (ISGT)*. Available at: https://www.eee.hku.hk/~hfwen/papers/ISGT-OPP-2013.pdf

[29] YU JJQ, LAM AYS & LI VOK. 2014. Chemical Reaction Optimization for the Set Covering Problem. *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*, p. 512–519. Available at: https://doi.org/10.1109/CEC.2014.6900233

[30] ZHOU M, CENTENO VA, PHADKE AG, HU Y, NOVOSEL D & VOLSKIS HAR. 2008. A preprocessing method for effective PMU placement studies. *3rd International Conference on Deregulation and Restructuring and Power Technologies, DRPT*.

**How to cite**

CARNIEL A & MESTRIA M. 2021. A chemical reaction optimization algorithm for phasor measurement unit placement. *Pesquisa Operacional*, **41**: e231454. doi: 10.1590/0101-7438.2021.041.00231454.