Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 112 (2017) 167–176

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Knowledge Based and Intelligent Information and Engineering Systems, KES2017, 6-8 September 2017, Marseille, France

Microgrid Components Clustering in a Digital Ecosystem Cooperative Framework

Khouloud Salameh[a,*], Richard Chbeir[a], Haritza Camblong[b], Ionel Vechiu[c]

[a] *Univ. Pau & Pays Adour, Anglet, 64000, France*
[b] *University of Basque Country, Donostia, 20018, Spain*
[c] *ESTIA Institute of Technology, Bidart, 64210, France*

Abstract

A MicroGrid (*MG*) is a distributed power system consisting of a number of heterogeneous components having direct/indirect impacts on each other. In order to provide an appropriate collaboration (from several perspectives) between components, we propose a "Digital Ecosystem Cooperative Framework" called *DECF*. In this paper, we present the clustering algorithm of *DECF* designed to build Alliances by gathering all the DE heterogeneous components having similar needs and preferences. Conducted simulations showed that the proposed algorithms yield significant results.

Keywords: Clustering; Digital Ecosystems; Microgrids; Cyber-Physical Systems; Information Processing

1. Introduction

A MicroGrid (*MG*) is a smaller version of the traditional power grid [10] which consists of a number of heterogeneous components (power generation, electrical loads, and storage systems) all within a controlled network. An *MG* can enhance the power reliability thanks to the local power generation and its ability to be islanded from the main grid. Thus, blackouts and power disturbances are significantly minimized. Since an *MG* is composed of a number of heterogeneous components, each having a direct/indirect impact on the other components and consequently on the entire environment, there is a need of establishing a dedicated internal *MG* cooperation addressing the components' heterogeneity and the problem of power exchange from different perspectives: technical [16], ecological [7] and economical [14]. In addition, the power exchange problem becomes more tricky with the rapidly growing population, the increasing energy demand, and the growing number of electrical equipment to be integrated into the *MG*. However, and to the best of our knowledge, none of the current approaches [16,20,13,12,7,14] seems to keep the pace since they don't consider the aforementioned perspectives at the same time nor allow end-users to fine-tune the importance of each one of them.

To address these issues, we propose *DECF*, a 'Digital Ecosystem Cooperative Framework' designed for optimizing the *MG* power exchange. *DECF* contains two main components: 1) the **Alliances Builder** provides an appropriate clustering algorithm aiming at gathering all the heterogeneous components having similar needs and preferences, and

* Corresponding author
E-mail address: khouloud.salameh@univ-pau.fr

1877-0509 © 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of KES International
10.1016/j.procs.2017.08.010

2) the **Seller2Buyer Matcher** is applied inside each cluster and between clusters, targeting a better collaboration inside the *MG*. In this paper, we detail the **Alliances Builder** of *DECF* and show how it meets the *MG* constraints.

The rest of the paper is organized as follows. Section 2 provides details about existing power exchange optimization techniques and their drawbacks. In Section 3, an overview of our *MG* information layers is given, before detailing and illustrating in Section 4 the clustering algorithm. In Section 5, the experiments conducted to validate our approach and the main results obtained are presented. Section 6 concludes the paper and draws several future directions.

2. Related work

Many approaches have been proposed in the literature to solve the optimization problem of the power exchange. They can be categorized into two main groups: game-theory based[16,12] and agent-based[1,7,11].

2.1. Game-theory approaches

In[16], the main goal was to develop an *MG* power exchange model which incorporates several energy sources (considered as Microgrids), allowing them to reduce the power load on the main grid and to minimize the transmission power losses over the distribution lines. In[12], the authors developed an approach that enables to determine the optimal operation of a solar-powered *MG* with respect to the consumers demands. The adopted scenario is a multiple sellers/buyers scenario, consisting of a village generating enough power and able to satisfy the demands (homes needs). The objective of the proposed approach is to make the village be at least cost-neutral in power while improving the revenue of the producers by comparing the uniform and discriminatory bidding. In[17], the authors developed a non-cooperative model within which the Plug-in Hybrid Electric Vehicles (PHEV) can decide on the amount of energy they want to sell to the main grid. In addition, the authors proposed a scheme for determining the trading price of the power exchanged between the PHEVs and the main grid.

2.2. Agent-based approaches

The literature is rich with examples of agent-based *MGs* optimization applications[1,7,11]. In most of these approaches[11], the *MG* is designed as a distributed power network comprising various distributed agents (generators, storage and loads, etc.) that are operated in interconnected or islanded mode. To do so, JADE framework is commonly adopted for agents' modeling. In[7], the authors developed a multi-agent system that aims to minimize *MG*'s photovoltaic (PV) operating costs and the toxic pollutants emissions while maximizing the output of the energy sources. In[1], a decentralized control architecture for *MG* was presented, aiming at maximizing the use of renewable energy sources and minimizing the use of conventional generators. The proposed control architecture contains different types of agents (such as PV agent, Fuel cell agent, etc.), where each represents a major component in the *MG*.

2.3. Discussion

None of the existing approaches can solve all the requirements presented previously. First, all of the existing approaches do not cope with ALL of the three objective aspects of an *MG*: technical, economical, and ecological aspects. Second, agent-based approaches[1,7,11] showed an efficiency in modeling all types of components, each represented by an agent, while game-theory approaches[16,12,17] failed in doing that by targeting solely the optimization of one type of *MG* components. Third, end-user requirements were almost absent in the existing approaches, with the exception of[7]. All that lead us to develop a new cooperative model, based on a solid information model, taking into account the various aspects of an *MG* while allowing the user to assign each aspect with an appropriate importance.

3. *MG* Information Layers

Beware that the *MG* can be perceived as a cyber-physical system, we designed our *MG* information model as a 5C architecture (Connection, Conversion, Cyber, Cognition and Configuration), complemented with additional modules specific to the needs in the *MG*. Our information modeling relies on three layers briefly described in what follows.

- Field Layer (FL): Via this layer, the data collector gathers all data exchanged between *MG* components via a low-level communication environment[19] relying on standardized protocols (e.g., BACnet[9] , Modbus[15] , etc.). Once gathered, those data are stored in a low-level data repository and pushed up to the next layers.
- Knowledge Layer (KL): In order to resolve the interoperability issues and open up the possibility to model the new trends in today's energy systems (i.e., prosumers, electric vehicle, etc.), it is essential to capture and understand the semantics of exchanged data to ensure a seamless communication between the *MG* components. Through this layer, the semantic middle-ware insures the semantic translation of the collected data using our ontology-based information model called *OntoMG*[18]. Furthermore, the reasoner is responsible of processing information and using it to infer additional value thanks to many rules and constraints defined in this layer.

- Management Layer (ML): In this layer, a collaborative diagnostics, a self-optimization for disturbance, and a remote visualization for the users (via an integrated simulation and synthesis) are provided. Besides, the information extracted from the knowledge layer is processed in order to achieve the objectives of the *MG*. To do so, a battery of advanced management services (e.g., Demand side management, minimization of transmission losses, etc.) is designed. The *DECF* framework belongs to those services and consists of two main modules, **Alliance Builder** and **Seller2Buyer Matcher**. The Alliance Builder component is provided to allow the *MG* to cluster its components according to several criteria as described in the following. The Seller2Buyer Matcher allows to align the schedule the collaboration between components.

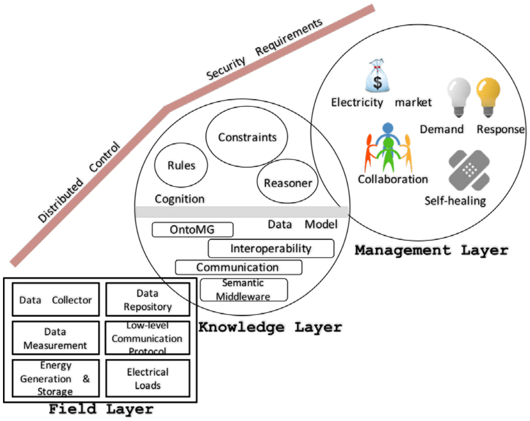


Figure 1. Our simplified *MG* information architecture

4. Alliance Builder

As it is more beneficial to promote an *MG* internal power exchange rather than relying on the main grid[16], our Alliance Builder has been designed to gather, into a set of alliances, the components having some interest to cooperate and exchange power in *MG* while taking into account important (technical, ecological and economical) aspects. In other words, an alliance consists of a number of *MG* components aiming at reducing internally the transmission power losses, polluting less, maintaining a power balance (between the generation and the consumption), ensuring a stable energy trading, etc. Since existing clustering techniques[5,6,3,2,8] cannot be adopted to cluster the *MG* components (mainly due to their heterogeneity), we propose our own algorithm. Before detailing the process, it is essential to present some formal definitions used in our study. Let us consider an *MG* consisting of K components which are heterogeneous (e.g., consumer, producer, etc.) and called nR.

Definition 1 (***MG* Component *[nR]***). *An MG component has the possibility to play one or several roles during its lifetime (i.e., produce, consume and store power). Formally, an nR is represented as:$\prec$ Id, Eco, Ecolo, Op, Geo, T $\succ$ where Id, Eco, Ecolo, Op, Geo represent its identification, economic, ecological, operational and geographical property sets respectively at a time T ∈ [1,...,H] / H =24, since we are studying the behavior of the MG in an interval of one hour. The property sets are defined in OntoMG (More details can be found in[18])*♦

Definition 2 (**Power Gap *[G]***). *A power gap defines the power surplus, need or satisfaction of a component or a set of components. Formally, a power gap of a component or a set of components $\mathcal{R}$, denoted as G, is defined as:*

$$G(\mathcal{R}) = \sum_{i=1}^{|\mathcal{R}|}(nR_i.g - nR_i.d + nR_i.s) \tag{1}$$

where $nR_i \in \mathcal{R}$, and g, d, and s are respectively the component power generation, demand, and storage ♦

Definition 3 (**Components *[nR^+], [nR^-], [nR^0])]***). *A seller, denoted as nR^+, has a power surplus ($G(nR) > 0$), while a buyer, denoted as nR^-, has a power need ($G(nR) < 0$). A self-satisfied , denoted as nR^0, has a power satisfaction ($G(nR) = 0$).* ♦

Definition 4 (**Alliance *[$\mathcal{A}$]* and Couple *[C]***). *An Alliance $\mathcal{A}$ is a set of at least one seller and one buyer having a mutual interest in working together. Formally, $\mathcal{A} :\prec \mathcal{R}^+, \mathcal{R}^- \succ$ where $\mathcal{R}^+ = \left(\bigcup_{i=1}^{n}\{nR_i^+\}\right)$, $\mathcal{R}^- = \left(\bigcup_{j=1}^{m}\{nR_j^-\}\right)$ and $(n+m) \leqslant K$, $n \geqslant 1$, and $m \geqslant 1$. A couple, denoted as C, is a special case of an alliance composed of only one buyer and one seller* ♦

Definition 5 (**Cost *[P]***). *The cost of one or several components is defined according to the costs related to its operational/technological, economical, and ecological properties. It represents the transmission power losses costs (operational), the power generation costs (economic) and the environmental impact costs (ecological) of the MG components during their functioning. Although it can be defined using different aggregation functions (e.g., maximum, average, etc.), we adopted the weighted sum function to combine the different objective aspects costs, allowing the user to tune the weight of each criterion in accordance with her priorities. Formally:*

$$P(\mathcal{R}, W) = w_{op} \times P_{op}(\mathcal{R}) + w_{eco} \times P_{eco}(\mathcal{R}) + w_{ecolo} \times P_{ecolo}(\mathcal{R}) \tag{2}$$

where $P_{op}(\mathcal{R})$ represents the operational cost of $\mathcal{R}$, $P_{eco}(\mathcal{R})$ represents the economic cost of $\mathcal{R}$, $P_{ecolo}(\mathcal{R})$ represents the ecological cost of $\mathcal{R}$, and $w_{op} + w_{eco} + w_{ecolo} = 1$ and $w_{op}, w_{eco}, w_{ecolo} \geqslant 0$♦

Note that, in this study, the cost P does not consider the power exchange cost. This latter is considered in the seller-to-buyer module. The main reason behind this choice, is building stable alliances based on reducing the ecological, economical and operational costs independently from any market tariffs changes. In addition, in this way we are prioritizing the alliances formation based on the three-dimensional factors without any market influence. For instance, let us consider a consumer $C1$ having a need of 200 KW, and two producers both able to satisfy the need of $C1$: a solar-powered system $S1$ with a tariff of 50 Euros and a diesel generator $S2$ with a tariff of 25 Euros. In our case, $C1$ and $S1$ will belong to the same alliance since we are privileging the ecological aspect on the market tariffs. However, while integrating the market prices, $C1$ and $S2$ will belong to the same alliance neglecting the ecological aspect.

Definition 6 (**Operational Cost** *[P_{op}]*). *The operational or technical cost of one or several components $\mathcal{R}$, denoted $P_{op}(\mathcal{R})$, is defined as:*

$$P_{op}(\mathcal{R}) = PWLossCost \times \sum_{i=0,j=0}^{n,m}(PWLoss_{i,j}) + PWasteCost \times (|G(\mathcal{R}) - \sum_{i,j}^{n,m}(PWLoss_{i,j})|) \tag{3}$$

where $\forall$ i and j, nR_i^+ and $nR_j^- \in \mathcal{R}$, and $n + m = |\mathcal{R}|$. The technical cost depends on various parameters such as the power losses $PWLoss(nR_i^+, nR_j^-)$ between the seller nR_i^+ and the buyer nR_j^-, and the fixed price $PWLossCost$ paid by the buyer nR_j^- per unit of power loss (e.g., 0.5 €/watt). In addition, the wasted power is calculated by subtracting the power lost in $\mathcal{R}$ from its gap G($\mathcal{R}$), and is multiplied by the fixed price $PWasteCost$ paid per unit of power wasted (e.g., 1 €/watt). Note that the power loss $PWLoss$ between two components is defined as follows:

$$PWLoss_{i,j} = R_{ij} \times I^2 + \beta \times Q_i \tag{4}$$

where R_{ij} is the resistance of the distribution line between the two components i and j, β is the fraction of power lost, I is the current flowing over the distribution line and Q_i represents the power flowing between the two components. ♦

Definition 7 (**Economical Cost** *[P_{eco}]*). *The economical cost of one or several components $\mathcal{R}$, denoted $P_{eco}(\mathcal{R})$, is defined as:*

$$P_{eco}(\mathcal{R}) = \sum_{i=0}^{|\mathcal{R}|}(nR_i.SUCost + nR_i.SDCost) + \sum_{j=0}^{n}(nR_j^+.PWCost \times nR_j^+.g) \tag{5}$$

where $\forall$ i, $nR_i \in \mathcal{R}$, and $\forall$ j, $nR_j^+ \in \mathcal{R}$, and $n \leqslant |\mathcal{R}|$. The economic cost depends on several factors such as the startup cost $SUCost$ and the shutdown cost $SDCost$ of each MG component nR_i in $|\mathcal{R}|$. In addition, it considers the production cost $PWCost$ paid by the seller nR_j^+ per unit of power production.♦

Definition 8 (**Ecological Cost** *[P_{ecolo}]*). *The ecological cost of one or several components $\mathcal{R}$, denoted $P_{ecolo}(\mathcal{R})$, is defined as:*

$$P_{ecolo}(\mathcal{R}) = GasEssCost \times \sum_{i=0}^{n}(nR_i^+.GasEss \times nR_i^+.g) \tag{6}$$

if $\forall$ i, $nR_i^+ \in \mathcal{R}$, and $n \leqslant |\mathcal{R}|$. The ecological cost depends on the toxic gas emissions $GasEss$ evolved during the power production, and the cost $GasEssCost$ per unit of gas emission. ♦

Definition 9 (**Isolated** *[$\mathcal{I}$]*). *An MG component nR is called isolated $R_{\mathcal{I}}$ if adding it to any existing alliance increases the cost of the alliance. Formally, an $nR \in R_{\mathcal{I}}$ if $P(\mathcal{A}' \cup \{nR\}) > P(\mathcal{A}')\ \forall\ \mathcal{A}' \in \left(\bigcup_{i=1}^{L} \mathcal{A}_i\right)$ where L is the number of created Alliances* ♦

Definition 10 (**Neighborhood** *[$\mathcal{V}$]*). *The neighborhood of a couple C, denoted $\mathcal{V}(C)$, is the set of one or more sellers or buyers, allowing the initial couple C to maintain its cost $P(C)$ after its/their integration. Formally,*

$$\mathcal{V}(C) = (\bigcup\{nR_i\})\ \text{if}\ \forall\ nR_i\text{:}\quad P(C) = \begin{cases} P(\{C.nR^+\}\bigcup\{nR_i\})\ \text{where}\ nR_i \in \left\{\bigcup_{j=1}^{m} nR^+\right\} \\ P(\{C.nR^-\}\bigcup\{nR_i\})\ \text{where}\ nR_i \in \{\bigcup_{k=1}^{n} nR^-\} \end{cases}$$

where n and m are the number of Sellers and Buyers, respectively. ♦

The pseudo-code of the Alliances Builder is provided in Algorithm 1. In short, the first phase of the process (lines 1-4) consists of taking away the self-satisfied components that have no need to sell or buy power in the *MG*. Then, a classification is done aiming at identifying the sellers and the buyers that are willing to enter the power exchange process. The idea implies that an *MG* component (a seller or a buyer) should join an alliance A rather than B, if it is able to decrease the cost of A $C(A)$ more than the cost of B $C(B)$. In other words, a component should be beneficial to the alliance, in that it should reduce its ecological, economical and operational cost to the maximum while reducing the wasted power into the alliance. To ensure that we are forming alliances with minimum costs, we start by selecting the couple (the seller and the buyer) having the minimum cost, and adding in the components that can reduce this cost. Once done, we move to the next couple having the next minimum. The complete process is explained in details in what follows. After this classification, a ‘start couple selection’ is initiated (line 9), resulting in one or more couples having the minimum cost, minimum gap and maximum number of neighbors. In the aim of encouraging the MG components cooperation, a compatibility test (line 15) is applied between the resulting start couple(s) and the existing alliances called ‘Candidate Alliances Selection’.It consists of selecting all the existing alliances that can reduce their costs by adding the start couples’ seller, buyer or the whole couple. When there is no resulting candidate alliance, we create a new alliance formed by the start couple. Then, a ‘Final candidate alliance’ (line 17) is achieved, consisting of creating an alliance formed by the candidate alliance having the biggest benefit by adding the start couple. After the creation of the alliance, this latter is updated by adding its neighbors able to reduce its costs. If none exists, a new ‘start couple selection’ is launched. The whole process is repeated until there is no more start couple. An overview of our Alliance Builder process is shown in Fig. 2. In the following, we detail its modules.

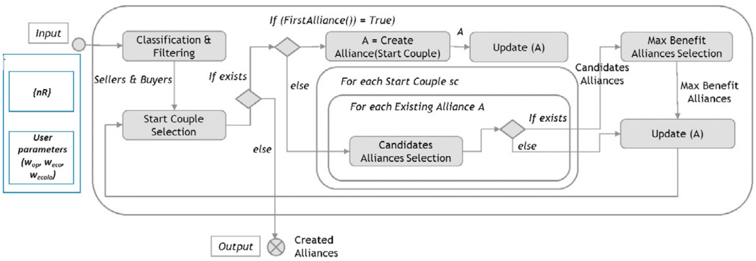


Figure 2. A simplified activity diagram of the Alliance Builder

Algorithm 1: Global Algorithm

Input: $nR[], W[]$ // Set of MG components and weights (operational, economical and ecological)
Output: $C\mathcal{M}$ // The classification of the Micogrid
1 $CC \leftarrow CC()$ // Retrieve the the set of MG components classified into three sets: sellers, buyers and self-satisfied
2 $nR^{+}[] \leftarrow CC.\mathcal{R}^{+}, nR^{-}[] \leftarrow CC.\mathcal{R}^{-}, nR^{0}[] \leftarrow CC.\mathcal{R}^{0}$ // Retrieve the Sellers, the Buyers, and the Self-satisfied
3 $\mathcal{R}_{\mathcal{A}}[] \leftarrow [], \mathcal{R}_{C}[] \leftarrow [], \mathcal{R}_{\mathcal{I}}[] \leftarrow [], \mathcal{R}_{C\mathcal{A}}[] \leftarrow []$ // Initialize a list of Alliances, Couples, Isolated, and candidate alliance
4 **while** $| nR^{+}[] | > 0$ & $| nR^{-}[] | > 0$ **do**
5 $\mathcal{R}_{C}[] \leftarrow StartCoupleSelection(nR^{+}[], nR^{-}[], \mathcal{R}_{\mathcal{A}}[], W[])$
6 **if** $| \mathcal{R}_{\mathcal{A}}[] | = 0$ **then**
7 $Couple \leftarrow RND(\mathcal{R}_{C}[])$
8 $Alliance \leftarrow UPDATE(nR^{+}[], nR^{-}[], Couple, W[])$ $\mathcal{R}_{\mathcal{A}}[] \leftarrow ADD(\mathcal{R}_{\mathcal{A}}[], Alliance)$
9 **else**
10 $\mathcal{R}_{C\mathcal{A}}[] \leftarrow CandidateAllianceSelection(nR^{+}[], nR^{-}[], \mathcal{R}_{C}[], W[])$
11 **if** $| \mathcal{R}_{C\mathcal{A}}[] | > 0$ **then**
12 $\mathcal{R}_{\mathcal{A}}[] \leftarrow FinalAllianceSelection(nR^{+}[], nR^{-}[], \mathcal{R}_{C\mathcal{A}}[], W[])$
13 **else**
14 $Couple \leftarrow RND(\mathcal{R}_{C}[])$
15 $Alliance \leftarrow UPDATE(nR^{+}[], nR^{-}[], Couple, W[])$
16 $\mathcal{R}_{\mathcal{A}}[] \leftarrow ADD(\mathcal{R}_{\mathcal{A}}[], Alliance)$
17 **for** *each* $nR_{i}^{+} \in nR^{+}[]$ **do**
18 $\mathcal{R}_{\mathcal{I}}[] \leftarrow ADD(\mathcal{R}_{\mathcal{I}}[], nR_{i}^{+})$ // This allows to add the isolated sellers to the isolated list
19 $nR^{+}[] \leftarrow REMOVE(nR^{+}[], nR^{+}[])$ // This allows to remove the isolated sellers from the sellers list
20 **for** *each* $nR_{i}^{-} \in nR_{i}^{+}$ **do**
21 $\mathcal{R}_{\mathcal{I}}[] \leftarrow ADD(\mathcal{R}_{\mathcal{I}}[], nR_{i}^{-})$ // This allows to add the isolated buyers to the isolated list
22 $nR^{-}[] \leftarrow REMOVE(nR^{-}[], nR^{-}[])$ // This allows to remove the isolated buyers from the buyers list
23 $C\mathcal{M}.\mathcal{R}_{\mathcal{A}} \leftarrow \mathcal{R}_{\mathcal{A}}[]$
24 $C\mathcal{M}.\mathcal{R}_{\mathcal{I}} \leftarrow \mathcal{R}_{\mathcal{I}}[]$
25 $C\mathcal{M}.\mathcal{R}^{0} \leftarrow nR^{0}[]$
26 **return** $C\mathcal{M}$ // Return the set of MG components, classified into three separate sets of alliances, isolated and self-satisfied components
27

Algorithm 2: Start Couple Selection

Input: $nR^{+}[], nR^{-}[], \mathcal{R}_{\mathcal{A}}[], W[]$ // Set of Sellers, Buyers, Created Alliances and weights (operational, economical and ecological)
Output: $\mathcal{R}_{C}[]$ // Set of Start Couples
1 $\mathcal{R}_{C}[] \leftarrow MCC(nR^{+}[], nR^{-}[], W[])$ // Select the couples having the minimum Cost
2 **if** $| \mathcal{R}_{C}[] | > 1$ **then**
3 $\mathcal{R}_{C}[] \leftarrow MGC(\mathcal{R}_{C}[])$ // Select the couples having the minimum Gap
4 **if** $| \mathcal{R}_{C}[] | > 1$ **then**
5 $\mathcal{R}_{C}[] \leftarrow MNC(\mathcal{R}_{C}[])$ // Select the couples having the maximum number of neighbors
6 **if** $| \mathcal{R}_{\mathcal{A}}[] | = 0$ **then**
7 $\mathcal{R}_{C}[] \leftarrow RND(\mathcal{R}_{C}[])$ // If many exist, choose a random start couple
8 **return** $\mathcal{R}_{C}[]$

4.1. Classification Module

This module consists of classifying the MG components into three separate sets: $(\{nR^+\})$, $(\{nR^-\})$, and $(\{nR^0\})$.

4.2. Start Couple Selection Module

The aim of this module is to select the starting couple(s) in each iteration. The pseudo-code of the Start Couple Selection is provided in Algorithm 2. It starts by selecting the couples having the minimum cost (line 1). If many resulting couples exist, the couple having the minimum gap is selected (lines 2-3). If many exist, the couple having the biggest number of neighbors is selected (lines 4-5). If many exist and if there is no existing alliance, the start couple is randomly chosen from the list of start couples (lines 6-7). Otherwise, the start couple will be a list of the couples having the biggest number of neighbors.

4.3. Candidate Alliances Selection Module

The goal of this module is to select the existing alliances, that are able to decrease their costs by integrating any of the start couples' seller, buyer or both. The pseudo-code of the Candidate Alliances Selection is provided in Algorithm 3. If there is no existing alliance, a new alliance is created by a random start couple and then updated with the function Update in the aim of testing the possibility to integrate its neighbors (lines 2-5). Otherwise, for each existing alliance having a $Gap > 0$ and for each start couple, it calculates the alliances costs after adding the start couple buyer and the whole couple (since it is unnecessary to add a seller to an alliance that already has a surplus of power). Here, if the new cost is less than the initial alliance cost, the new alliance is added to the list of candidate alliances (lines 9-19). The same test is done on the alliances having a $Gap < 0$, by testing the new alliances costs after adding the start couple's seller or the whole couple (lines 20-30). Algorithm 4 shows the pseudo-code of the update function.

4.4. Final Alliance Selection Module

The goal of this module is to select the final candidate alliance which has the biggest cost reduction when adding the start couple seller, buyer or the whole couple. The pseudo-code of the Final Alliance Selection is provided in Algorithm 5. For each candidate alliance resulting from the 'Candidate Alliances Selection' module, it selects the alliances having the maximum benefit (line 2). If there is no resulting alliance, a new alliance is created formed by a random start couple (lines 3-6). Otherwise, only one will be chosen randomly (lines 8-9). Then, a selection of a new start couple selection is done. The whole same process will be repeated until there is no more start couples. An illustration is provided in Appendix A.

Algorithm 3: Candidate Alliances Selection

```
   Input: nR+ [], nR− [], R_C [], R_A [], W[]   // Set of Sellers, Buyers, Start Couples, Created Alliances and weights (operational, economical and ecological)
   Output: R_CA []                               // Set of Candidate Alliances
1  R_CA [] ← []
2  if |R_A[]| = 0 then
3      C ← RND(R_C[])                            // If many exist, choose a random start couple
4      nR+ [] ← REMOVE(nR+ [], C.R+)
5      nR− [] ← REMOVE(nR− [], C.R−)
6      A ← UPDATE(nR+ [], nR− [], C)             // Test if we can add in any start couple neighbor
7      R_A[] ← ADD(R_A[], A)                     // Add the new alliance to the created alliances set
8  else
9      for each C_i ∈ R_C[] do
10         for each A_j ∈ R_A[] do
11             if Gap(A_j) > 0                   // If A_j needs a buyer
12             then
13                 A_k ← ADD(A_j, C_i.R−)
14                 if P(A_k, W[]) ⩽ P(A_j, W[])   // If the couple's buyer C_i.R− reduces the cost of A_j
15                 then
16                     A_j ← A_k                 // Add the buyer C_i.R− to the alliance A_j
17                     R_CA[] ← ADD(R_CA[], A_j) // Add the alliance A_j to the set of candidate alliances R_CA[]
18                 else if P(ADD(A_k, C_i.R+, W[])) ⩽ P(A_j, W[])   // If the couple's buyer C_i.R− and the couple's seller C_i.R+ reduce the cost of A_j
19                 then
20                     A_j ← ADD(A_k, C_i.R+)    // Add the seller C_i.R+ to the alliance A_j
21                     R_CA[] ← ADD(R_CA[], A_j) // Add the alliance A_j to the set of candidate alliances R_CA[]
22             else if Gap(A_j) < 0              // If A_j needs a seller
23             then
24                 A_k ← ADD(A_j, C_i.nR+)
25                 if P(A_k, W[]) ⩽ P(A_j, W[])   // If the couple's seller C_i.nR+ reduces the cost of A_j
26                 then
27                     A_j ← A_k                 // Add the seller C_i.nR+ to the alliance A_j
28                     R_CA[] ← ADD(R_CA[], A_j) // Add the alliance A_j to the set of candidate alliances R_CA[]
29                 else if P(ADD(A_k, C_i.R−, W[])) ⩽ P(A_j, W[])   // If the couple's buyer C_i.R− and the couple's seller C_i.R+ reduce the cost of A_j
30                 then
31                     A_j ← ADD(A_k, C_i.R−)    // Add the buyer C_i.R− to the alliance A_j
32                     R_CA[] ← ADD(R_CA[], A_j) // Add the alliance A_j to the set of candidate alliances R_CA[]
33 return R_CA[]
```

Algorithm 4: Update

Input: $nR^{+}[], nR^{-}[], C, W[]$ // Set of Sellers, Buyers, the Couple to be updated and the set of weights (operational, economical and ecological)
Output: $\mathcal{A}$ // The resulting alliance

Alliance $\mathcal{A}$
Set of candidate Alliance $\mathcal{R}_{\mathcal{CA}}[]$
Set of final Alliance $\mathcal{R}_{\mathcal{FA}}[]$
$V_C \leftarrow \mathcal{V}(C)$
if $|V_C| \geqslant 1$ // If C has neighbors
then
  if $Gap(C) > 0$ // If C needs a buyer
  then
    for *each* $nR_i \in V_C$ **do**
      $R[] \leftarrow Add(C, nR_i)$
      if $Gap(nR_i) < 0$ & $P(R[], W[]) \leqslant P(C)$ // If nR_i is a buyer and reduces the cost of C
      then
        $\mathcal{A} \leftarrow ADD(C, nR_i)$ // Creating an alliance $\mathcal{A}$ resulting from adding the neighbor nR_i to the couple C
        $nR^{-}[] \leftarrow REMOVE(nR^{-}[], nR_i)$ $\mathcal{R}_{\mathcal{CA}}[] \leftarrow ADD(\mathcal{R}_{\mathcal{CA}}[], \mathcal{A})$ // Add the alliance $\mathcal{A}$ to the set of candidate alliances $\mathcal{R}_{\mathcal{CA}}[]$
    $\mathcal{R}_{\mathcal{FA}}[] \leftarrow MCAB(\mathcal{R}_{\mathcal{CA}}[])$ // Select the alliances having the biggest benefit
    if $|\mathcal{R}_{\mathcal{FA}}[]| > 1$ **then**
      $\mathcal{A} \leftarrow RND(\mathcal{R}_{\mathcal{FA}}[])$
  if $Gap(C) < 0$ // If C needs a seller
  then
    for *each* $nR_i \in V_C$ **do**
      $R[] \leftarrow Add(C, nR_i)$
      if $Gap(nR_i) > 0$ & $P(R[], W[]) \leqslant P(C)$ // If nR_i is a seller and reduces the cost of C
      then
        $\mathcal{A} \leftarrow ADD(C, nR_i)$ // Creating an alliance $\mathcal{A}$ resulting from adding the neighbor nR_i to the couple C
        $nR^{+}[] \leftarrow REMOVE(nR^{+}[], nR_i)$ $\mathcal{R}_{\mathcal{CA}}[] \leftarrow ADD(\mathcal{R}_{\mathcal{CA}}[], \mathcal{A})$ // Add the alliance $\mathcal{A}$ to the set of candidate alliances $\mathcal{R}_{\mathcal{CA}}[]$
    $\mathcal{R}_{\mathcal{FA}}[] \leftarrow MCAB(\mathcal{R}_{\mathcal{CA}}[])$ // Select the alliances having the biggest benefit
    if $|\mathcal{R}_{\mathcal{FA}}[]| > 1$ **then**
      $\mathcal{A} \leftarrow RND(\mathcal{R}_{\mathcal{FA}}[])$
return $\mathcal{A}$

Algorithm 5: Final Alliance Selection

Input: $nR^{+}[], nR^{-}[], \mathcal{R}_C[], \mathcal{R}_{\mathcal{CA}}[], W[]$ // Set of Sellers, Buyers, Start Couples, Candidate Alliances and weights (operational, economical and ecological)
Output: $\mathcal{R}_{\mathcal{A}}[]$ // Set of Created Alliances

Set of Alliances $\mathcal{R}_{FA}[]$
$\mathcal{R}_{\mathcal{FA}}[] \leftarrow MCAB(\mathcal{R}_{\mathcal{CA}}[])$ // Select the alliances having the biggest benefit
if $|\mathcal{R}_{\mathcal{FA}}[]| = 0$ **then**
  $C \leftarrow RND(\mathcal{R}_C[])$
  $nR^{+}[] \leftarrow REMOVE(nR^{+}[], C.\mathcal{R}^{+})$
  $nR^{-}[] \leftarrow REMOVE(nR^{-}[], C.\mathcal{R}^{-})$
  $\mathcal{A} \leftarrow UPDATE(nR^{+}[], nR^{-}[], C, W[])$ // Update the created alliance
  $\mathcal{R}_{\mathcal{A}}[] \leftarrow ADD(\mathcal{R}_{\mathcal{A}}[], \mathcal{A})$ // Add the alliance to the existing alliances
else
  $\mathcal{A} \leftarrow RND(\mathcal{R}_{\mathcal{FA}}[])$
  $\mathcal{R}_{\mathcal{A}}[] \leftarrow ADD(\mathcal{R}_{\mathcal{A}}[], \mathcal{A})$ // Add the alliance to the existing alliances
return $\mathcal{R}_{\mathcal{A}}[]$

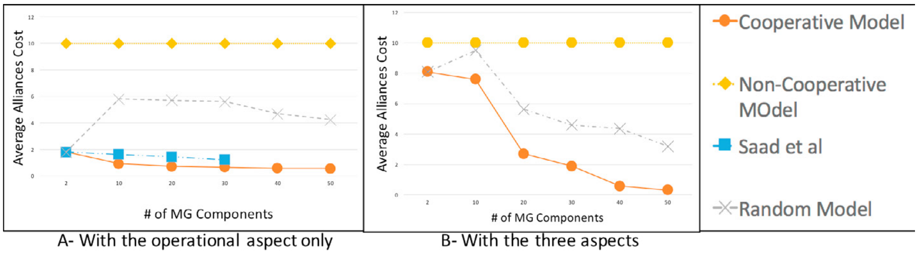
5. Experiments

We have developed a prototype to validate our *DECF* framework. A set of tests have been conducted to validate our approach as explained below. A prototype has been implemented using Java to conduct the test on a PC with an Intel Core i7-3630 QM CPU, 2.40 GHz processor with 8 GB RAM. Since the *MG* is relatively a recent concept in the power systems area, there is a lack of a current Benchmark to be based on. Hence, we carried out our experimental scenario inspired by the one provided in [16] but adapted to fit better the scope of our study. Here, we set up an *MG* within an area of 10 km × 10 km with: 1) the main grid located at the onshore , and 2) the *MG* components randomly located within this area. The power gap (G) of any *MG* component nR: 10 MW $\leqslant G(nR) \leqslant$ 316 MW. Note that, the exchange cost between an *MG* component and the Main Grid is set to 10. The main criteria used to evaluate the effectiveness of our approach are: i) the alliances formation impact on the *MG* operation, and ii) the time needed to generate the alliances. We detail the obtained results below.

5.1. Alliance Builder impact on the MG operation

We measured the alliances costs per *MG*, where we varied the number of components from 2 to 50 components (following the recommendation in [4], in an average of 10 times. Note that, the highest number of components used in the literature was 30 components [16]. In this test, four different scenarios were considered: 1) a non-cooperative one, consisting of calculating the average cost of the *MG* components exchange with the Main grid, 2) a random one, consisting of calculating the costs average of a random alliances formation, 3) the cooperative model presented in [16], and 4) our cooperative one.

As mentioned before, the work in [16] takes only into account the operational aspect of the *MG*. Hence, in order to be able to compare their approach with our work, we considered only the operational aspect in our cost calculation (by assigning 1 to the operational cost weight and 0 to the others, i.e., $w_{op} = 1$ and $w_{eco} = w_{ecolo} = 0$). Fig.



A- With the operational aspect only B- With the three aspects

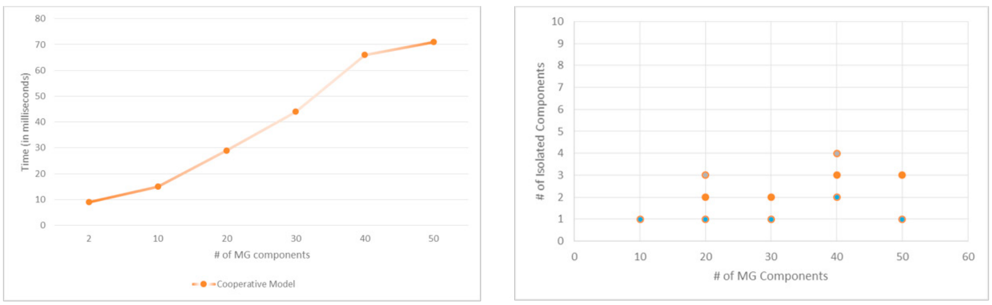
Figure 3. Average Alliances Cost w.r.t. the number of *MG* components

3-A shows that, the worst case scenario is the non-cooperative one, with a constant value of 10. This result shows that as the number of components increases, the resulting alliances average cost decreases more the random scheme averages. This is due to the fact that, for our cooperative algorithm, as N increases, it becomes easier for the *MG* components to find other components with which they can cooperate in a beneficial way in order to decrease the alliances costs and therefore increase the performance of the *MG*. In addition, it is clear that, compared to the random scenario, our proposed method has a significant performance improvement, in terms of average alliances cost, which is increasing with N and reaching up to 40% of cost reduction (at N = 50) relative to the random scenario. Comparing to the existing approach in[16], obtained results show that our proposed model ensures better results reaching up to 30% of cost reduction (at N = 30). For the rest of the tests, we reconsider the three aspects of the *MG* equally ($w_{op} = w_{eco} = w_{ecolo} = 3.33$). In Fig. 3-B, we show the same test conducted while integrating all the aspects in the cost computation. The result shows that our method is better than the other approaches as well.

Another test was conducted to calculate the resulting noise of the alliance builder module. It consisted of calculating the number of the generated isolated components in the Classified Microgrid ($\mathcal{CM}$). Fig. 4-B shows that the biggest number of isolated comes down to ”4”, which can be considered as a very promising result and fully satisfies our initial goal in conceiving a cooperative environment.

5.2. Alliance Builder Performance

In addition to testing the effectiveness of our approach in reducing alliances costs, we evaluated its time performance. This test consisted of measuring the time to build the alliances while varying the number of *MG* components. Fig. 4-A shows that the time needed to create alliances grows in an almost linear fashion (since N is small) w.r.t. the number of components. This is intuitive, since every component is a part of the alliances builder algorithm input and therefore it should be parsed in order to associate it the adequate alliance.



A- Time performance w.r.t. the number of MG components B- Isolated components w.r.t. the number of MG components

Figure 4. Time performance & Noise of Alliance building

6. Conclusion

In this work, we proposed a digital ecosystem cooperative framework for *MG* distribution network. The proposed approach is based on two main modules: 1) the alliances builder and 2) the Seller2Buyer matcher. In this paper, we detailed our novel clustering algorithm consisting of gathering the *MG* components into ‘alliances’. Each alliance contains a number of *MG* components, having mutual interests. Their interests is expressed by an objective function, taking into account several *MG* objectives: operational, economical and ecological. Simulation results show that the proposed algorithm yields a significant improvement leading to minimizing the total power exchange cost in the *MG*. We are currently integrating our framework into ISare project [1] in order to test its performance in real situations. Also, we are exploring different alternatives to automatically compute the weights provided by end-users.

References

1. H. N. Aung, A. Khambadkone, D. Srinivasan, and T. Logenthiran. Agent-based intelligent control for real-time operation of a microgrid. In *Power Electronics, Drives and Energy Systems (PEDES) & 2010 Power India, 2010 Joint International Conference on*, pages 1–6. IEEE, 2010.
2. M. Berry. *Survey of Text Mining : Clustering, Classification, and Retrieval*. Springer, September 2003.
3. M. W. Berry and M. Castellanos, editors. *Survey of Text Mining II: Clustering, Classification, and Retrieval*. Springer, London, 2008.
4. J. Eto and et. al. Overview of the certs microgrid laboratory test bed. In *Integration of Wide-Scale Renewable Resources Into the Power Delivery System, 2009 CIGRE/IEEE PES Joint Symposium*, pages 1–1. IEEE, 2009.
5. J. A. Hartigan. Clustering algorithms. 1975.
6. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.
7. X. Jia and e. al. Application of multi-agent technology in micro-grid system. In *Advanced Power System Automation and Protection (APAP), 2011 International Conference on*, volume 2, pages 962–967. IEEE, 2011.
8. L. Kaufman and P. J. Rousseeuw. *Finding groups in data : an introduction to cluster analysis*. Wiley series in probability and mathematical statistics. Wiley, New York, 1990. A Wiley-Interscience publication.
9. G. M. Kim, S. H. Kim, S. H. Hong, and J. Lee. Design of a bacnet-zigbee gateway for smart grid in buildings. In *Conference Anthology, IEEE*, pages 1–5. IEEE, 2013.
10. R. H. Lasseter. Microgrids. In *Power Engineering Society Winter Meeting, 2002. IEEE*, volume 1, pages 305–308. IEEE, 2002.
11. T. Logenthiran, D. Srinivasan, and D. Wong. Multi-agent coordination for der in microgrid. In *Sustainable Energy Technologies, 2008. ICSET 2008. IEEE International Conference on*, pages 77–82. IEEE, 2008.
12. I. Maity and S. Rao. Simulation and pricing mechanism analysis of a solar-powered electrical microgrid. *Systems Journal, IEEE*, 4(3):275–284, 2010.
13. F. A. Mohamed and H. N. Koivo. Online management of microgrid with battery storage using multiobjective optimization. In *Power Engineering, Energy and Electrical Drives, 2007. POWERENG 2007. International Conference on*, pages 231–236. IEEE, 2007.
14. A.-H. Mohsenian-Rad and el. al. Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid. *Smart Grid, IEEE Transactions on*, 1(3):320–331, 2010.
15. D.-g. Peng, H. Zhang, L. Yang, and Y. Li. Design and realization of modbus protocol based on embedded linux system. In *Embedded Software and Systems Symposia, 2008. ICESS Symposia'08. International Conference on*, pages 275–280. IEEE, 2008.
16. W. Saad, Z. Han, and H. V. Poor. Coalitional game theory for cooperative micro-grid distribution networks. In *Communications Workshops (ICC), 2011 IEEE International Conference on*, pages 1–5. IEEE, 2011.
17. W. Saad, Z. Han, H. V. Poor, and T. Başar. A noncooperative game for double auction-based energy trading between phevs and distribution grids. In *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*, pages 267–272. IEEE, 2011.
18. K. Salameh and et. al. A generic ontology-based information model for better management of microgrids. In *Artificial Intelligence Applications and Innovations*, pages 451–466. Springer, 2015.
19. V. K. Sood, D. Fischer, J. Eklund, and T. Brown. Developing a communication infrastructure for the smart grid. In *Electrical Power & Energy Conference (EPEC), 2009 IEEE*, pages 1–7. IEEE, 2009.
20. A. G. Tsikalakis and N. D. Hatziargyriou. Centralized control for optimizing microgrids operation. In *Power and Energy Society General Meeting, 2011 IEEE*, pages 1–8. IEEE, 2011.

Appendix A. Alliances Builder Illustration

Let us consider an *MG* consisting of 9 components having the power generation *(g)*, demand *(d)* and storage *(s)* as shown in Table A.1.

- Classification module: After classifying the *MG* components, they will be put into three main categories: the sellers willing to sell their power surplus ($nR_1 \rightarrow nR_1^+$, $nR_4 \rightarrow nR_2^+$, $nR_6 \rightarrow nR_3^+$, $nR_8 \rightarrow nR_4^+$), the buyers willing to buy their power needs ($nR_2 \rightarrow nR_1^-$, $nR_3 \rightarrow nR_2^-$, $nR_5 \rightarrow nR_3^-$, $nR_9 \rightarrow nR_4^-$), and the self-satisfied components ($nR_5 \rightarrow nR_1^0$) (c.f. Table A.2.). In what follows, the sellers will be visually represented by '$\triangledown$' and the buyers by '$\vee$'.
- Start Couple Selection module: After classifying the *MG* components, a start couple selection is executed. Table A.3. shows the costs matrix of all the couples. Table A.5. shows the execution result of the Start Couple Selection algorithm. After selecting the couples having the minimum cost, several resulting couples having $Cost = 2$ (line 2) are found, the couple having the minimum gap is selected (line 3). In this example, there are many resulting couples having the same gap $Gap = 2$ (line 4), hence, the couple having the maximum number of neighbors is selected (line 5). Also here, several couples have the same number of neighbors $\mid \mathcal{V}(C) \mid= 2$,

[1] http://www.i-sare.net/html5/index.html

thus, a random couple is selected (e.g., $C_1 =< nR_2^-, nR_3^+ >$). As a result, a new alliance A_1 is created consisting of the seller and the buyer of the start couple, $A_1 :< \{nR_2^-\}, \{nR_3^+\} >$ as shown in Fig. A.5-1.

- Update Module: After creating the first alliance $A_1 =< \{nR_2^-\}, \{nR_3^+\} >$, the update module is called to test the possibility of adding any of its neighbors as long as it reduces the alliance cost. $\mathcal{V}(A_1) = \{nR_4^+\}$ and $C(ADD(A_1, nR_4^+)) < C(A_1)$, hence the alliance should be updated by adding the seller nR_4^+ resulting $A_1 =< \{nR_2^-\}, \{nR_3^+, nR_4^+ >\}$ as shown in Fig. A.5-2.
- Candidate Alliance Selection Module: After creating the alliance $A_1 =< \{nR_2^-\}, \{nR_3^+, nR_4^+\} >$, we update the costs matrix by removing the buyers and the sellers being part of existing alliances (cf. Table A.4). Here, a new start couple selection is done, resulting the couple $C_2 :< nR_3^-, nR_1^+ >$ as shown in Fig. A.5-3.

Component	*Generation (g)*	*Demand (d)*	*Storage (s)*
nR_1	17	0	0
nR_2	2	35	1
nR_3	4	10	1
nR_4	20	0	5
nR_5	5	5	0
nR_6	0	0	3
nR_7	6	20	0
nR_8	5	3	1
nR_9	10	20	5

Table A.1. Example of 9 components

Component	(g)	(d)	(s)	$Gap(G)$
nR_1 (nR_1^+)	17	0	0	+17
nR_2 (nR_1^-)	2	35	1	-32
nR_3 (nR_2^-)	4	10	1	-5
nR_4 (nR_2^+)	20	0	5	+25
nR_5 (nR_1^0)	5	5	0	0
nR_6 (nR_3^+)	0	0	3	+3
nR_7 (nR_3^-)	6	20	0	-14
nR_8 (nR_4^+)	5	3	1	+3
nR_9 (nR_4^-)	10	20	5	-5

Table A.2. Classification module result

	nR_1^-	nR_2^-	nR_3^-	nR_4^-
nR_1^+	7.5	6	1.5	6
nR_2^+	3.5	10	5.5	10
nR_3^+	14.5	1	5.5	1
nR_4^+	14.5	1	5.5	1

Table A.3. Cost Matrix of the couples

Before creating a new alliance with the resulting start couple, we check the possibility of adding any of the couple's seller, the buyer or both to the existing alliance A_1 (cf. Table A.2.6). Our example shows the impossibility of adding the buyer nR_3^- or the seller nR_1^+ or the whole couple (line 10) to the existing alliance (line 11) $A_1 =< \{nR_2^-\}, \{nR_3^+, nR_4^+\} >$, since once added to A_1, they would increase its cost (lines 27-33). Hence, a new alliance $A_2 =< \{nR_3^-\}, \{nR_1^+\} >$ is created. Here, a new start couple selection is done, since this alliance A_2 has no neighbors to check the possibility of adding it resulting the couple $C_3 :< nR_1^-, nR_2^+ >$. Before creating a new alliance with the resulting start couple, we check again the possibility of adding any of the couple's seller, buyer or both to the existing two alliances A_1 and A_2. Our example shows that by adding the seller and the buyer of the couple to the alliance A_2, the cost of this latter is decreased. Hence, the alliance A_2 will be updated and becomes $A_2 =< \{nR_1^+, nR_2^+\}, \{nR_3^-, nR_1^-\} >$. Here, the only one remaining component, nR_4^-, will be an isolated since it is impossible to add it to the existing alliances as it increases their costs once added as shown in Fig. A.5-4.

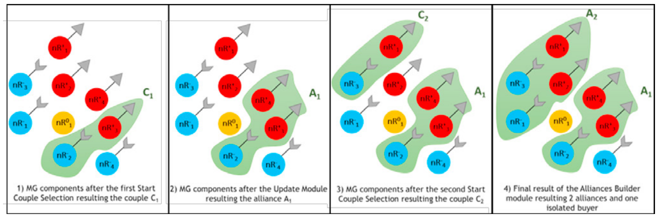


Figure A.5. Visual representation of the *MG* components' clustering status after each step

	nR_1^+	nR_2^+
nR_1^-	7.5	3.5
nR_3^-	1.5	5.5
nR_4^-	6	10

Table A.4. Updated Cost matrix of the remaining couples

Line	*Result*
Line 1	$\mathcal{R}_C = \{< nR_2^-, nR_3^+ >, < nR_4^-, nR_3^+ >, < nR_1^-, nR_4^+ >, < nR_4^-, nR_4^+ >\}$
Line 2	*True*
Line 3	$\mathcal{R}_C = \{< nR_2^-, nR_3^+ >, < nR_4^-, nR_3^+ >, < nR_1^-, nR_4^+ >, < nR_4^-, nR_4^+ >\}$
Line 4	*True*
Line 5	$\mathcal{R}_C = \{< nR_2^-, nR_3^+ >, < nR_4^-, nR_3^+ >, < nR_1^-, nR_4^+ >, < nR_4^-, nR_4^+ >\}$
Line 6	*True*
Line 7	$\mathcal{R}_C =< nR_2^-, nR_3^+ >$

Table A.5. Start Couple Selection algorithm execution

Line	*Result*
Line 2	*False*
Line 9	$< nR_3^-, nR_1^+ >$
Line 10	$< nR_2^-, nR_3^+, nR_4^+ >$
Line 11	*False*
Line 22	*True*
Line 27	*False*
Line 29	*False*
Line 33	$\mathcal{R}_{C\mathcal{A}} = []$

Table A.6. Candidate Alliance Selection algorithm Execution example