Revista de Informática Teórica e Aplicada - RITA - ISSN 2175-2745
Vol. XX, Num. XX (2018) 11-XX

RESEARCH ARTICLE

Multi-Agent System Integrated to Power Systems Analysis Tool for Self-Healing in Smart Distribution Systems

Sistema Multiagente Integrado à Ferramenta de Análise de Sistems de Potência para Autorrecuperação em Sistemas Inteligentes de Distribuição

Kelly S. S. Costa[1]*, Filipe Saraiva[1]

Abstract: This paper describes an application built from the integration between two frameworks, Python for Power System Analysis (PyPSA) and Python Agent DEvelopment (PADE) whose purpose is to solve the problem of self-healing in electrical distribution systems. For that, this system simulates a smart grid, adding an information layer to the conventional electrical distribution system that allows the detection of interruptions in the energy supply. The proposal detects the affected area, the tie lines available to reestablish the energy supply, performs the make decision algorithm to select the best network configuration — based on the comparison obtained by calculating the power flow performed in each possible configuration, and make the reconfiguration of the network, reestablishing the power supply. The algorithm is performed in an automatic, distributed, and real-time way, by the use of multi-agent technology. The proposal was tested in 33-bus power distribution systems topology with different scenarios. The results obtained show the proposal works as expected, reducing the power losses after the network restoration.
Keywords: multi-agent system — smart grids — energy supply — self-healing

Resumo: Este artigo descreve uma aplicação construída a partir da integração entre dois frameworks, Python for Power System Analysis (PyPSA) e Python Agent DEvelopment (PADE), cujo objetivo é solucionar o problema de autorrecuperação em sistemas de distribuição elétrica. Para isso, este sistema simula uma rede elétrica inteligente, adicionando uma camada de informação ao sistema elétrico de distribuição convencional que permita a detecção de interrupções no fornecimento de energia. A proposta detecta a área afetada, as *tie lines* para restabelecimento do fornecimento de energia, realiza a tomada de decisão para selecionar a melhor configuração de rede – baseada na comparação obtida pelo cálculo do fluxo de potência realizado em cada configuração possível, e realiza a reconfiguração da rede, restabelecendo o fornecimento de energia. O algoritmo é realizado de maneira automática, distribuída e em tempo real, a partir do uso da tecnologia de sistemas multiagentes. A proposta foi testada na topologia de sistemas de distribuição de energia de 33 barramentos em diferentes cenários. Os resultados obtidos mostram que a proposta funciona como o esperado, reduzindo as perdas elétricas após a restauração da rede.
Palavras-Chave: sistemas multiagentes — redes elétricas inteligentes — fornecimento de energia — autorrecuperação

[1] *Faculty of Computer Science, Institute of Exact and Natural Sciences, Federal University of Pará – Belém - PA, Brazil*
***Corresponding author**: kellydosocorro@gmail.com
DOI: http://dx.doi.org/10.22456/2175-2745.XXXX • **Received:** dd/mm/yyyy • **Accepted:** dd/mm/yyyy

1. Introduction

In the current context of society, the relevance of the electric power system is undeniable. This is because such systems are strongly rooted in the activities of modern society, which was developed by the use of electricity. The discovery and consequent success in electricity control enabled several inventions that guaranteed for humanity comfort, practicality, agility, among other benefits.

Electric Power Systems can be defined as a set of physical equipment and elements of electrical circuits combined, which act in a coordinated way in order to generate, transmit and distribute electric energy to its consumers [1]. Unlike other systems such as gas or water systems, there is not a storage of energy in an electric power system, so the energy is generated on demand, implying that system is a real-time one. As a real-time system, there are a lot of challenges in its maintenance, as well as, a series of limitations in its constitution, establishing,

therefore, a vast area of studies that aim at the improvement and resolution of its problems.

In this way, more and more research is conducted in order to ensure quality, sustainability and modernization of these systems. Thus, the smart grids arise in this scenario with the purpose of inserting, side-by-side to the power grid, an information and communication system which allows the monitoring of this network and therefore, the detection of potential problems, the automation of decision-making, and their execution in the face of problems that arise, minimizing costs related to maintenance and the system's own operation, among other possibilities [2].

As stated, smart grids have a great potential to modernize and help with some issues with the current electrical power system. In smart grid scenario, artificial intelligence has the power to improve the decision making which such systems should perform. In this way, this paper presents an application which simulates the functioning of a smart grid in order to solve the self-healing problem, which consists in finding the sequence of switch operation to reach an optimal restoration state according to some specified objective [3]. This ability can be achieved with two different approaches, centralized or decentralized.

In the centralized approach, it is necessary to have prior knowledge of the topology, as well as the addresses of all the devices that make up the network, because, as the name indicates, a single component will be responsible for tracking and monitoring changes in the network. With this perspective, works with mathematical models such as [4] which proposed a method that uses mixed integer linear programming (MILP) and nonlinear programming (NLP) to resolve and adjust respectively. On the other hand, in the decentralized approach, this need does not exist, since this technique has its origins in distributed systems, the components are running on different devices, interconnected by a network, communicating through the exchange of messages, as an example, this work [5], that uses a decentralized coordinated energy management system (EMS) to connect the micro grids which coexist as independent systems.

Thus, the decentralized approach was chosen by the author, based on researches [2] that suggests the use of Multi-Agent Systems (MAS) as an approach to the problem of self-healing. So, this paper presents an application which proposes to detect and automate the recovery of areas affected by supply interruptions. For that, this application was based on the algorithm proposed by [6], which introduces a new approach with a reactive MAS for self-recovery in smart grids. This paper aims to present a new point of view for [6], adding a "global" approach which performs the calculation of power flow in all the network, in order to avoid that the final choice of agents cannot turn out to be a false positive.

For simulation purpose, the IEEE 33-bus network model was used while for the analysis of the results, logs are used to record the final configuration of the network after all the process of fault detection, delimitation of the affected area, detection of viable configurations, decision making and network reconfiguration, being possible to observe, therefore, the new topology obtained by the interaction between the agents that compose the system.

Then, this paper splits into another 6 sections, as described bellow:

The second section, introduces the description of the problem and the different approaches already used for its solution.

The third section exposes the concept of multiagent systems and bring examples of its use in different scenarios.

The fourth section describes how the multiagent systems approach has been used to solve the problem of self-healing in smart grids.

The fifth section aims to present the characteristics of the network topology chosen for the testing stage.

The sixth section presents simulated scenarios and the results of the application against the tests performed.

Finally, the last section presents the general review and discussion of the results, as well as the conclusion of this work.

2. Problem Description and Literature Review

The self-healing is a problem that occurs in a network in the face of a power outage and is nothing more than the capacity of an electrical network to reconfigure itself, in order to restore the electricity supply interrupted by an eventual supply failure. Thus, it is an important issue of power distribution systems due to the importance of the electric power in the modern society.

Currently, the electric power it is a base of many systems, devices and services that make the human life better in many aspects, whether these complexes as enabling the World Wide Web operation, or simpler, such as turning on a lamp. Therefore, an interruption in the distribution of this resource is capable of harming the quality of life of its consumers. In this cases, in conventional electrical networks, there is not another possibility than human interference to solve this matter manually, making the necessary adjustments and maintenance.

However, regarding smart grids — that is, modern electrical networks that have intelligent characteristics, such as monitoring the state of the network and making decisions that imply the quality and functioning of distribution — the self-healing is an extremely desired ability [7] and complex capacity, given its distributed nature and changes in the network structure caused by the power flow calculation to reach the optimal result. In practice, this problem usually is divided into two steps, the first occurs when the system detects that one or more loads have suffered from supply interruption, then, once the lack of energy is identified, the system should look for a solution that aims to soften or totally restore the affected area and reconfigure itself, this last one, being the second step.

As an important problem of smart grids, the problem of self-healing over the years has become the source of several researches which aim to present different perspectives for the implementation of this functionality. Some of then, as [8] that uses hybrid techniques, fuzzy to fault diagnosis step and Ant Colony Optimization (ACO) to reconfiguration. Another example, is the work [9], which present an distributed solution, similar to the technique of multi agents, that uses individualized switching that cooperate with adjacent switches to reach an self-healing plan.

In this work, the Multi-Agents System (MAS) approach was the chosen technology by the authors, based on the algorithm proposed by this paper [6], chosen in such a way of validate the solution presented in [2]. In the MAS proposed in this paper, each component of the grid (loads and lines) has a linked agent that represents themselves in the system. These agents have the ability to monitor and record the activities of their components as well as interfere in their activities, closing and opening keys as required by the algorithm. For this algorithm development, beyond the load and switch agents representing loads and lines respectively, there are another 2 type of agents, these being: sharing and network, each of these, carrying an administrative responsibility of the system, both involving a common area in which agents must at some point register, so that the algorithm can proceed with its steps, as explained in future sections. As a basis, naturally this work has many similarities with [6], but, there are some differences between this 2 approaches, in particular, two big differentials, one being the existence of a network agent and the second the purpose of this agent being necessary in the system.

Thus, it is understood that this work is limited to presenting a solution to the problem of self-recovery in smart grids using a multi-agent system, using the global scope of the network, in order to obtain a new strategy for the proposed algorithm. The next two sections will present the characteristics of multi-agent systems and the the application of this approach to self-healing.

3. Multi-Agent Systems (MAS)

According with [10], a Multi-Agent System (MAS) is a system composed of several agents, an environment, a set of possible interactions and, possibly, at least one organization, such as the equation 1. So, this kind of system consists of a configuration made by autonomous decision-making agents that exists in an environment where each agent can act based on a set of specified behaviours, interact between themselves through the exchange of messages and collect information about the environment. A representation of this agent can be seen in the figure 1.

$$MAS = Agents + Environment + Interactions + Organization \quad (1)$$

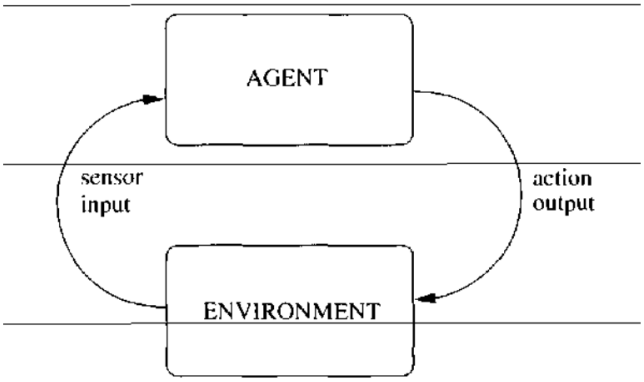

Figure 1. The agent and its interaction with the environment in which it is inserted [12]. The sensor (input) is the agent's ability to perceive changes in the environment. The action (output) is the way in which this agent infringes modifications to the environment in which it is inserted.

3.1 Behaviors

As mentioned earlier, each agent of the system has a limited set of actions to perform in the face of the various scenarios inflicted on the environment that contains it. This set of actions its called "behaviours", and, different from the functions of the Object-oriented Paradigm, the agent has the autonomy to invoke such behaviors as necessary, in such a way that their execution isn't always sequential. Even in the face of an unknown scenario, the agent must have a course of action, what makes the agent fault tolerant, an indispensable requirement inside the MAS.

3.2 Communication

Regarding the communication, it can be said that the agents have a communication that avoids simple interaction, the communication between the agents has a purpose, very similar to human communication, in which [11] classifies them as negotiation, coordination, cooperation and teamwork.

3.3 Distributed System

As it's a system where its components are distributed in a computer network, in which processing competition occurs, [12] classifies MAS as a subclass of distributed systems, given its skills of synchronization and coordination of agents activities in run time, and also, the agent behaviour of been primarily concerned with themselves, even though they are acting on behalf of some user/owner.

3.4 Multi-agent systems applications in Power Systems

Given its characteristics, the multi-agent systems technique has already been used for the construction of several applications, such as [13] where the authors also focus in fault detection, reconfiguration and restoration, but based their work on a proto-type circuit, the Circuit of the Future (CoF) and use the Java Agent Development Framework together with Matlab.

Another application of the MAS was proposed by [14] where they aim to develop a decentralized prototype of protection to distributed generation based on multi-agent that can detect the occurrence of High Impedance Fault (HIF), fault location and load shedding. For this, the agents interact themselves performing tasks of protection through autonomy and cooperating. As stated, the MAS approach has many characteristics that make it suitable for the development of tools aimed at the decentralized management of multiple devices, an important property for the area of power systems.

4. Multiagent Proposal

As well as exposed in the previous section, the nature of multi-agent systems presents several desirable characteristics to achieve this paper goal, that would be:

- A distributed environment representing the smart grid network;
- Agents that can be modeled to symbolize specific components of this smart grid, such as the load and the line;
- Customizable behaviours modeled so that they can occur at different stages.

Having proven the favorable nature of the MAS approach, it remains the implementation of an algorithm that represents this particular ability of the smart grid, the self-healing. Thus, the algorithm proposed follows practically the same steps as proposed by the [6] modifying some steps and introducing a new agent, the network agent. the algorithm proposed in and described below:

4.1 Self-healing with Multi-agent system

To understand how it works, then it is divided into stages, which take place sequentially in order to solve the proposed problem.

4.1.1 Making connections

In principle, all agents initialized are recognized by an agent called Agent Management System (AMS), responsible for managing all agents, who, in general, check their existence or not in the system. Here, each line and load is linked to an agent, so, each one of this agents send a message to Network agent (Fig. 2), an agent that in the future will perform the power flow. In order for the connections between loads and lines to be established, the agents of each line will also exchange messages with their respective load agents and both will store information regarding their neighbor and the line that connects them. Once connected, they know the basic information of their neighbors, however, they do not have full knowledge about the network.

4.1.2 Failure detection

In the system, the power supply is represented by a simple file that every load agent publishes as soon as it is created and therefore, it's necessary to delete the files of the agents

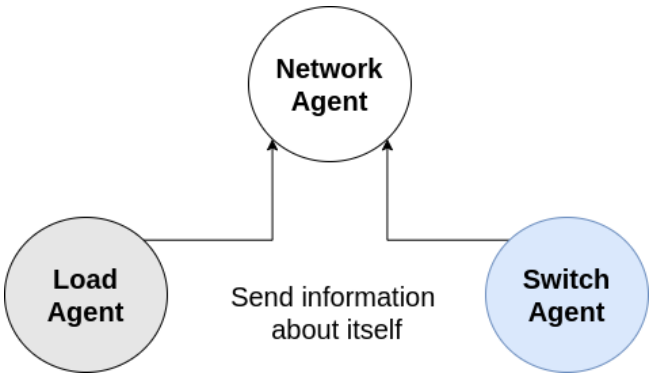

Figure 2. All the agents that represent loads and lines (switches) send their information to Network Agent.

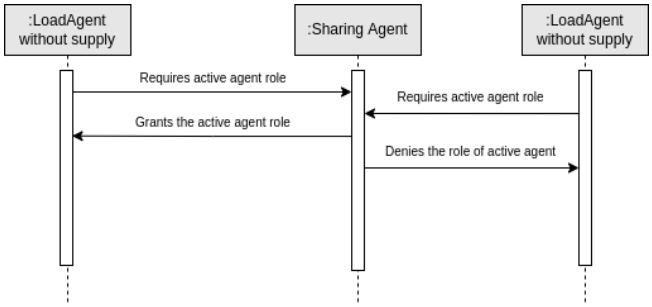

Figure 3. The Load agents in fault try to take the role of Active agent, but this role will only be assigned to one of them, which will be responsible for managing the mapping of the affected area.

to simulate the power supply failure. Once the absence of power is detected by the sensor of each agent affected by the fault, each load agent will send a message to Sharing Agent, but only the first message received will guarantee the sender, the power of an Active Agent, this is, the power of managing the self-healing process (Fig. 3). For other Load Agents who failed to register, the role remains as Passive Agents, that in practice will return the information as requested to solve the problem.

4.1.3 Mapping of affected area

In this step, the Active Agent asks the situation about the energy supply to its neighbors, who in turn, request their neighbors and so on, recursively. The information is collected and returned to the Active Agent as soon as the point of failure is found (Fig. 4), that is, the agent that does not have power supply, but is connected to an agent that does. Once the point of failure is found, the agents return their responses recursively to their requesting neighbor, and finally, for the Active Agent, while the agent that has power supply but is connected to the affected area, detects and communicates with the Line Agent that joins it to the area, requesting that this line is disabled. It is important to note that, in this mapping, the deactivated lines (tie lines) that can be used in the next stage are returned

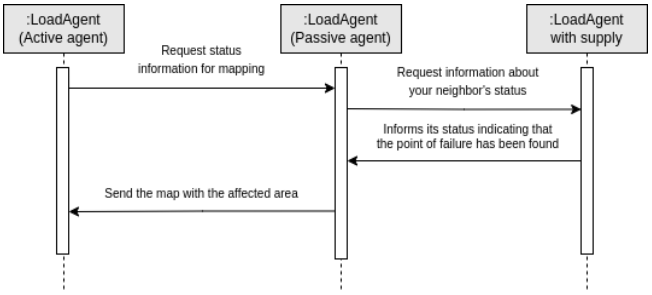

Figure 4. Along the requests for the mapping, a list containing those involved, whether loads, switches or tie lines is generated and is recursively returned to the active agent as soon as the failure point is found.

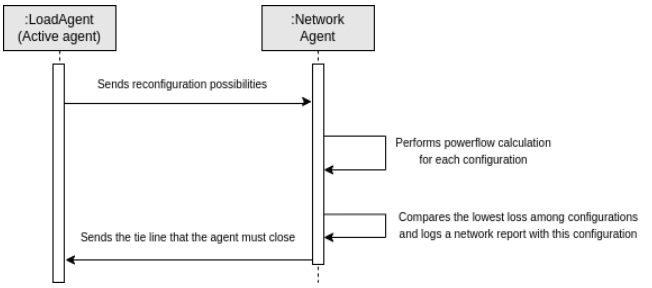

Figure 5. Network Agent from overview will run powerflow.

to the requester, thus, in possession of the affected area, the Active Agent will start the process of finding the best possible reconfiguration.

4.1.4 Detection of possible configurations

To detect the possible configurations, the active agent observes the number of inactive lines that were returned in the mapping, as these lines are the lines considered ”reserves” and that can be used to reestablish the energy supply for that area — in an eventual case where there isn't reconnection possibility, the algorithm is interrupted without being able to recover itself. After that, the Active Agent asks the network agent to calculate the power flow for each possible configuration, that is, testing each tie line as if it were active on the network (Fig. 5). At this moment, the network agent performs the due calculations taking into account the entire network, which differs from the algorithm proposed by [6] that performs these calculations only in relation to the affected area. Another significant point is that at this step, in the original algorithm, is carried out entirely under the responsibility of the Active Agent, differing from this work, which presents Network Agent as responsible for the calculations.

4.1.5 Network reconnection

At this point, the network agent will choose the best configuration by the results of the power flow. The chosen one will be the one who presents the lesser power loss. From this result, the Network Agent sends the tie line which presented the lowest result to Active Agent. Once the message is received, the Active Agent requests the closing of this line and the update of the Load Agents involved in the affected area (Fig. 6). After that, the affected agents update their data about neighbors and lines and recreate their files indicating the restoration of the power supply.

As a result of this steps, the electrical network is then reconfigured so that the loads previously affected can receive energy again. Next, the results of the simulations carried out will be presented.

5. Power System Topology

To prepare this work, it was necessary to choose a topology model in order to provide a test environment for the algorithm to be tested. Thus, for the purpose of comparing the results, the 33-bus model was chosen. This is distributed in such a way that a node is considered a source node responsible for supplying electrical energy to the other nodes of the topology.

5.1 33-bus

33-bus it is an official topology presented by IEEE. In this topology, only one of the nodes is a power supply, while the others just depend on the energy shared from this. The table 1 show the configuration used in this topology for the tests.

In this topology, in accordance with what was proposed in [15], 5 tie lines were used, with the following configurations:

Table 2. Tie lines info

Line	From	To	Resistance	Reactance
32	7	20	2.0	2.0
33	8	14	2.0	2.0
34	11	21	2.0	2.0
35	17	23	0.5	0.5
36	24	28	0.5	0.5

These last ones, start in the system as open switchs which are closed as required by the self-healing algorithm.

6. Simulations and Results

This section aims to present the technologies that enabled the construction of the algorithm, the scenarios that served for its validation and the results obtained.

As illustrated in the previous sections, the self-healing algorithm was developed using the concept of multi-agent systems and using the 33-bus network topology. In order for its construction and testing to be possible, some tools and simulation scenarios were needed, which are described in the following subsections.

6.1 Technologies utilized

For this application development, two frameworks were used using the Python programming language, these being Python Agent DEvelopment (PADE) and Python for Power System Analysis (PyPSA), the first one to modeling the MAS, while

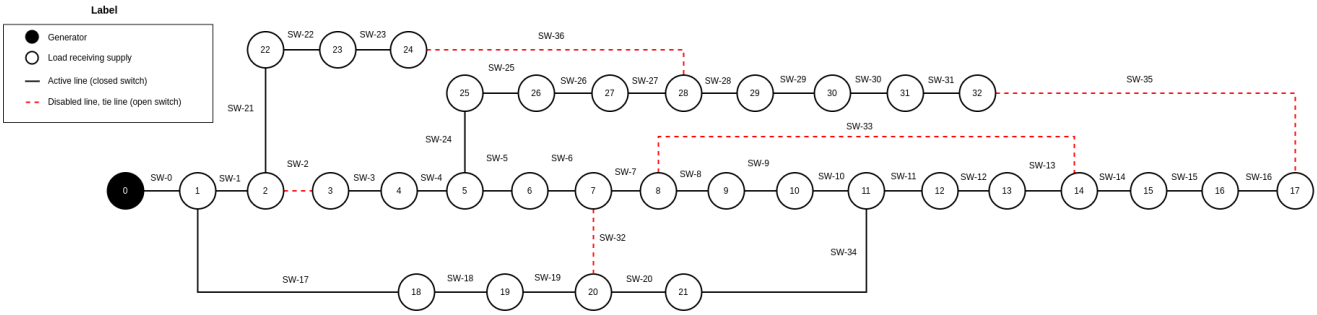

Figure 6. Example of reconnected network based on algorithm. The point of failure occurred on line (2, 3) and the tie line chosen for correction was line (11, 21).

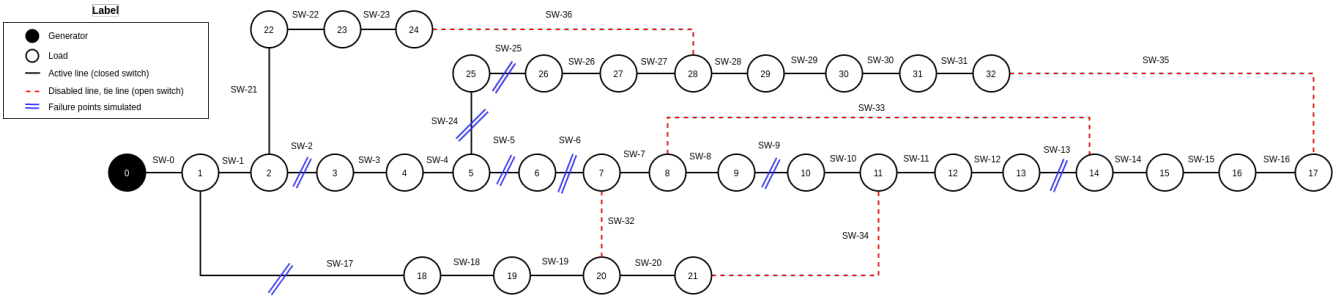

Figure 7. The 33-bus topology with tie lines and simulation scenarios.

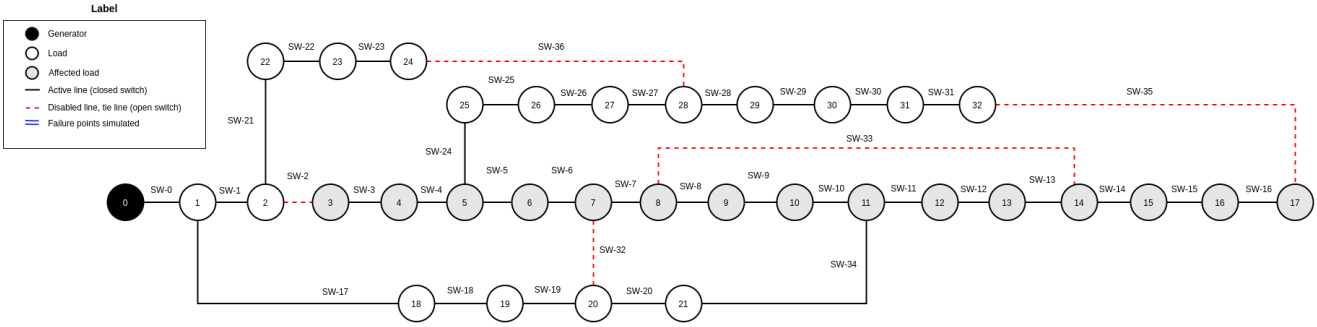

Figure 8. Fault between loads 2 and 3 (line SW-2). After the execution of the algorithm, the SW-2 line was deactivated and the SW-34 tie line connecting load 11 to 21 was closed.

Table 1. 33-bus lines info

Line	From	To	Resistance	Reactance
0	0	1	0.0922	0.0477
1	1	2	0.493	0.2511
2	2	3	0.366	0.1864
3	3	4	0.3811	0.1941
4	4	5	0.819	0.707
5	5	6	0.1872	0.6188
6	6	7	1.7114	1.2351
7	7	8	1.03	0.74
8	8	9	1.04	0.74
9	9	10	0.1966	0.065
10	10	11	0.3744	0.1238
11	11	12	1.468	1.155
12	12	13	0.5416	0.7129
13	13	14	0.591	0.5260
14	14	15	0.7463	0.545
15	15	16	1.289	1.721
16	16	17	0.732	0.574
17	1	18	0.164	0.1565
18	18	19	1.5042	1.3554
19	19	20	0.4095	0.4784
20	20	21	0.7089	0.9373
21	2	22	0.4512	0.3083
22	22	23	0.898	0.7011
23	23	24	0.896	0.7091
24	24	25	0.203	0.1034
25	25	26	0.2842	0.1447
26	26	27	1.059	0.9337
27	27	28	0.8042	0.7006
28	28	29	0.5075	0.2585
29	29	30	0.9744	0.963
30	30	31	0.3105	0.3619
31	31	31	0.341	0.5302

the last one, to simulate the power distribution network and its components. These technologies will be better described below.

PADE is a framework developed by the Group of Research/development in Intelligent Power and Energy Systems from the Federal University of Ceará (GREI/UFC) and Laboratory of Applied Artificial Intelligence from the Federal University of Pará (LAAI/UFPA). According to [16], this tool has a clear and easy approach that enables the creation, execution and monitoring of MAS in a distributed system. This tool provides a development environment to create custom agents and custom behaviours, this last one based on predefined types, for instance, cyclic behaviours that must keep executing while the agent exists in the system and also, some FIPA protocols such as Request Protocol, described in details at [17].

The other framework used in this work is Python for Power System Analysis (PyPSA) a framework developed with the aim of providing modern power system modeling. This tool is maintained by the Energy System Modelling group at the Institute for Automation and Applied Informatics at the Karlsruhe Institute of Technology (KIT) and in this project it is used with the purpose of simulating the electrical network and performing power flow calculations to obtain the best possible configuration of the network. But, more than the features mentioned, this tool provides many other features, which allow its use for various types of simulation of electrical networks, being able to take snapshots of the system at a given moment and having several models, among them, meshed multiply-connected AC and DC networks [18].

6.2 Simulation Scenarios

To test the algorithm, some failure scenarios were created from the 33-bus topology. In the figure 7 it is possible to see such situations, in such a way that highlighted in blue, are all the simulated failure points and highlighted in red, all the possible tie lines to use in case of failure. For the simulations, only points with a chance of self-healing were chosen in order to discover the behavior of the network with the increase in the number of hits and, consequently, of the tie lines involved. Such scenarios are summarized in the table 3.

Table 3. Failure simulations used

Simulation	Failure line	Number of affected nodes	Involved tie lines
1	(13, 14)	4	(8, 14), (17, 32)
2	(9,10)	8	(8, 14), (17, 32), (11, 21)
3	(5, 6)	12	(8, 14), (17, 32), (11, 21)
4	(2, 3)	15	(20, 7), (17, 32), (11, 21)
5	(1, 18)	4	(7, 20), (11, 21)
6	(25, 26)	7	(17, 32), (24, 28)
7	(6, 7)	11	(7, 20), (17, 32), (11, 21)
8	(5, 25)	8	(17, 32), (24, 28)

As shown in table, 8 failure scenarios were applied to collect results from this approach. Such scenarios were chosen based on the 33-bus simulations of the work [15] in order to obtain data that can compare both algorithms.

Thus, according to the data in the table, the simulation that will involve the highest involvement of loads is the simulation number 4, interrupting the supply of 15 loads while the one with the least involvement will be the first. Next, the results for each scenario discussed.

6.3 Results

Applying the algorithm under the scenarios previously described, at first, it was observed that there was no case in which the system could not recover itself.

6.3.1 Scenario 1

In this case, the fault occurs between the loads 13 and 14, in the line 14, as points the table 1 in the section 5. There are two possible tie lines to reconnect the network, the lines that connects (8, 14) and (17, 32).

Of the 5 attempts performed, in 100% of cases the system calculated the line (8, 14) as the best choice.

Table 4. Average time by simulation

Simulation	Failure line	Closed tie line	Time (ms)
1	(13, 14)	(8, 14)	6215.2
2	(9,10)	(11, 21)	9085
3	(5, 6)	(11, 21)	4492.4
4	(2, 3)	(11, 21)	6458
5	(1, 18)	(7, 20)	5569.6
6	(25, 26)	(24, 28)	4260.2
7	(6, 7)	(11, 21)	8265.6
8	(5, 25)	(24, 28)	4406.6

Failure line: represents the line where the failure occurs; **Closed tie line:** it is the tie line that was chosen for self-healing; **Time:** it is the average time in milliseconds to do all the self-healing process.

6.3.2 Scenario 2

This scenario the fault occurs between 9 an 10, having as available tie lines: (8, 14), (17, 32), (11, 21). During the 5 attempts, in all the simulations the chosen line was (11, 21).

6.3.3 Scenario 3

In this simulation, the chosen point of failure was between the loads 5 and 6. Then, the possibilities for this area points to tie lines (8, 14), (17, 32), (11, 21). In this case, an interesting scenario happened: not all of the attempts chosen the same tie line. Just one attempt select the tie line (7, 20) while in the others, the chosen one was (11, 21). This can occurs because the power loss difference of the tie line (11, 21) surpasses (7, 20) by 0.005MW.

6.3.4 Scenario 4

For this case, the failure is in the line among 2 and 3, where the picked tie line was also (11, 21) in 100% of the attempts.

6.3.5 Scenario 5

This scenario has the failure point at (1, 18). Similar to what happened in the scenario simulation 6.3.3, there is only one attempt that choose another line besides (7, 20). In one case, the chosen tie line was (11, 21), by a difference of 0.005MW between the power loss of them.

6.3.6 Scenario 6

The next case, the failure is among loads 25 and 26, and as result, select the tie line (24, 28) as the most viable reconnection candidate.

6.3.7 Scenario 7

In this scenario, the failure point simulated was between 6 and 7 and results in a 100% of the attempts in tie line (11, 21).

6.3.8 Scenario 8

In this last case, the failure point was the line that connects the loads 5 and 25. For the self-healing and reconnection of the network, the line chosen by the algorithm in all cases was the tie line between 24 and 28.

As noted, in some simulations there were some variations between the attempts, which have no impact on the algorithm's efficiency results, since the main objective remains to make it autonomous, in such a way that given a failure scenario, the best configuration of network can be established for reconnection, according to the data scanned by the agents at the time.

Thus, in all tested scenarios, the system was able to detect the failure, those involved, the possible network configurations and reconnect the affected individuals. This, of course, taking into account the presence of tie lines to meet such demands, still requiring human intervention in cases where no configuration is available.

From these results, it can be inferred from the patterns that the simulations that caused the longest execution time of the algorithm were those whose contemplate the following 3 conditions:

- The affected area was larger;
- There are more options of tie lines;
- The point of failure is located more in the center of the topology.

Because of this three items, the simulations 2 and 7 can be considered the worst-case scenario for the algorithm. Regarding the time obtained, this is superior in relation to the work [15] simulations, which is considered normal, given that the process is applied on a larger scale. A table reveals this comparison below:

Table 5. Time comparasion between approaches

Simulation	This work (ms)	Original work (ms)
1	6215.2	662
2	9085	579
3	4492.4	621
4	6458	660
5	5569.6	555
6	4260.2	586
7	8265.6	605
8	4406.6	-

Note: in the original work, the last one tested was not tested in a regular topology like the others, that is, in a 33-bus topology that has never been reconfigured.

7. Conclusions

This article, through the implementation of an approach previously proposed by another author, aimed to ratify his approach and increase it in order to cover the problem from another perspective. Thus, it is concluded that the implemented algorithm was able to reproduce and answer the self-healing problem in electrical power networks, using multi-agent systems and perform the network reconnection to a configuration whose losses are minimal, in a relatively longer time than the original algorithm, but still considered satisfactory, considering

that currently this procedure is manually – which demands much more time and also the involvement of humans for this operation.

Therefore, this article reinforces that the originally proposed idea can be validated, that is, the implementation of the self-healing feature of a smart grid, as well as generating other future works, involving more functionalities of the electrical power system that can be automated and implemented with artificial intelligence techniques.

References

[1] BENEDITO, R. A. de S. *Introdução a Sistemas Elétricos de Potência*. Disponível em: ⟨http://paginapessoal.utfpr.edu.br/raphaelbenedito/sistemas-eletricos-de-potencia-i/aulas/⟩.

[2] SARAIVA, F. d. O. *Aplicações Híbridas entre Sistemas Multiagentes e Técnicas de Inteligência Artificial para Redes Inteligentes de Distribuição de Energia Elétrica*. 2015. Dissertação (Doutorado em Engenharia Elétrica) — Universidade de São Paulo. Programa de Pós-Graduação em Engenharia Elétrica, São Carlos, SP, Brasil, 2015. Disponível em: ⟨https://www.teses.usp.br/teses/disponiveis/18/18154/tde-06062016-094659/pt-br.php⟩.

[3] OUALMAKRAN, Y.; MELENDEZ, J.; HERRAIZ, S. Self-healing for smart grids: Problem formulation and considerations. *3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, 2012.

[4] CAVALCANTE, P. L. et al. Centralized self-healing scheme for electrical distribution systems. *IEEE TRANSACTIONS ON SMART GRID*, 2016.

[5] WANG, Z. et al. Networked microgrids for self-healing power systems. *IEEE TRANSACTIONS ON SMART GRID*, 2016.

[6] CAMPOS, I. R. da C. Sistema multiagente reativo aplicado ao problema de autorrecuperação em smart grids. *Revista de Informática Teórica e Aplicada*, v. 27, n. 2, p. 127–139, 2020.

[7] BROWN, R. E. Impact of smart grid on distribution system design. *Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century, 2008 IEEE*, 2008.

[8] EL-HAMED, M. Z. A.; EL-KHATTAM, W.; EL-SHARKAW, R. Self-healing restoration of a distribution system using hybrid fuzzy control/ant-colony optimization algorithm. *3rd International Conference on Electric Power and Energy Conversion Systems*, 2013.

[9] TORRES, B. S.; FERREIRA, L. R.; AOKI, A. R. Distributed intelligent system for self-healing in smart grids. *IEEE Transactions on Power Delivery*, 2018.

[10] DEMAZEAU, Y. From interactions to collective behaviour in agent-based systems. 1995.

[11] PADGHAM, L.; WINIKOFF, M. *Developing Intelligent Agent Systems: A practical guide*. [S.l.]: John Wiley & Sons Ltd, 2004.

[12] WOOLDRIDGE, M. J. *An introduction to multiagent systems*. [S.l.]: JOHN WILEY & SONS, LTD, 2002. ISBN 0-471-4969I-X.

[13] NARESHKUMAR, K. et al. Application of multi-agents for fault detection and reconfiguration of power distribution systems. *IEEE Power Energy Society General Meeting*, 2009.

[14] XIANGJUN, Z. et al. Multi-agents based protection for distributed generation systems. *4 IEEE Intemational Conference on Electric Utility Deregulation, Restructuring and Power Technologies*, 2004.

[15] CAMPOS, R. d. C. *Aplicação de sistemas multiagentes ao problema de autorrecuperação em sistemas elétricos de distribuição do tipo smart grid*. 2018. Monografia (Bacharel em Ciência da Computação), UFPA (Universidade Federal do Pará), Belém, Brasil.

[16] MELO, L. S. et al. Python-based multi-agent platform for application on power grid. *International Transactions on Electrical Energy System*, n. 29, 2019. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/abs/10.1002/2050-7038.12012⟩.

[17] FIPA. *Stantard Status Specifications*. 2002. Disponível em: ⟨http://fipa.org/repository/standardspecs.html⟩.

[18] BROWN, T.; HÖRSCH, J.; SCHLACHTBERGER, D. Pypsa: Python for power system analysis. *Journal of Open Research Software*, v. 6, n. 4, 2018. Disponível em: ⟨https://doi.org/10.5334/jors.188⟩.