# Distributed Constraint Optimization Problem Solving in Unstable Environments

Saeid SamadiDana
*Department of Computer Science and IT*
*Austin Peay State University*
Clarksville, TN, USA
samadidanas@apsu.edu

***Abstract*—Distributed systems are widely used to share tasks in various systems which communicate to each other. one main reason to use these systems is their ability to keep each system's privacy and share only the required information. The success of these systems relies on robust communications among their nodes. Advances in network and communication technologies have led to a more robust quality solution for distributed problems as these systems heavily rely on network robustness and stability. Despite this progress, the communication problems such as delay, loss, and noise still exist in many environments that have dramatically affected the quality of distributed problem solutions. In some recent studies, these issues have been explored partially; however, the need to investigate these issues' impact specifically when combined seems necessary. This article studies the effect of message loss while there is a chance of distortion in the receiving messages. To have a better view of communication issues, both static and dynamic problems are tested. Three distributed algorithms, Distributed Stochastic Algorithm (DSA), Distributed Breakout Algorithm (DBA), and Max-Gain Message algorithm (MGM), are chosen to be tested in these environments, and their performance has been compared to each other. Test results show that all three algorithms are highly impacted by network instability, while DSA provides better results in general.**

***Keywords*—Multi-Agent System, Distributed, Problem, Dynamic, Noise, Optimization, Communication, Privacy**

## I. Introduction

Recent advances in network technology have provided a relatively robust environment for many distributed systems. Especially the 5-G, despite its challenges, seems will provide a more promising network reliability (Andrews et al., 2014). This progress can connect many heterogeneous devices and systems possible through the Internet of Things (Kiran, 2019; Mira, 2019). Nonetheless, in many cases having a robust and reliable network is difficult. The existence of distributed systems relies on communication among various parts (nodes) of them. Without a trustworthy network, distributed systems cannot exchange information promptly. For various reasons, a problem might be distributed over a network of solvers that work together to solve a problem. The need to maintain the privacy of individual systems or limit the amount of information shared or the computing power is why one may use distributed systems rather than centralized ones. Some of the main problems that distributed systems have solved are constraint satisfaction problems (CSP) and constrain optimization problems (COP). Distributed supply change managements(Alemany, Esteso, Ángel Ortiz, & del Pino, 2021) and various versions of path-finding(Yao, Qi, Wan, & Liu, 2019; Samadidana, Paydar, & Jouzdani, 2017) are just two examples of thousands of such problems. Generally, distributed algorithms share the tasks among various individual systems known as nodes (or agents). Each node is responsible for solving the local problem while communicating with other system nodes to solve the whole problem. Hence, the key to having a robust solution is exchanging the most up-to-date information in a timely manner. However, even a reliable network might face noise that can adversely impact the distributed algorithms' performance. Delay in communication and loss of messages are very common issues. Several studies such as (Samadidana, 2019; Wahbi & Brown, 2014) have investigated issues such as communication delay and loss and explored the effect of various sources of information stagnancy.

In this study, the performance of three distributed algorithms used to solve distributed constraint optimization problems (DCOP) has been analyzed. It should be noted that this work extends the findings presented at IEEE CCWC 2021 conference (Samadidana, 2021). To have a better understanding of the communication issues, both static and dynamic DCOP problems are tested. Moreover, this paper investigates the effect of communication delay and loss on these problems while there are some chances that the delivered messages are noisy and contain invalid information. To the author's best knowledge, this work is the first study that considers the combination of various sources of information stagnancy in DCOP problems. This study aims to advance the research community's knowledge of information stagnancy in distributed problem-solving. The algorithms that are used in this paper are DSA (Zhang, Wang, & L.Wittenburg, 2002), DBA (Yokoo & K.Hirayama, 1996), and MGM (Maheswaran, Pearce, & D, 2004). These algorithms are chosen as each represents a different approach to solve the DCOP problems. However, in all of them, the nodes communicate the latest information with each other. The structure of this paper is as follows: In Section II, Distributed Constraint Optimization Problems (DCOP) and Dynamic DCOP (DynDCOP) problems are introduced. Section III describes the three distributed algorithms. Test results are explained in Section IV. This section is divided into two subsections to investigate both static(Subsection IV-A) and dynamic problems (Subsection IV-B). Finally, in Section V the study is concluded, and the results are summarized.

## II. Definitions and Background

### A. Constraint Satisfaction Problems (CSP)

A Constraint Satisfaction Problem(CSP) is described as following (Russell & Norvig, 2020):

- A set of $n$ variables: $\mathbf{V} = \{v_1, \ldots, v_n\}$.
- Discrete, finite domains for each variable: $\mathbf{D} = \{\mathbf{D_1}, \ldots, \mathbf{D_n}\}$
  where $\mathbf{D}_i = \{d_{i,1}, \ldots, d_{i,j}\}$ and each $d_{i,j}$ is an element of domain $\mathbf{D_i}$
- a set of constraints $\mathbf{C} = \{c_1, \ldots, c_m\}$ where each $c_h(d_{i,1}, \ldots, d_{i,j} | 1 \leq i, j \leq n))$ is a function $c_h : D_{i,1} \times \cdots \times D_{i,j} \rightarrow \{true, false\}$.

Based on the definition, $D_{i,j}$ is a pair of domain values that have been assigned to variables $i$ and $j$. In addition, a constraint can be set either to true or false. The goal of the problem is to satisfy all the constraints or to conclude that no solution exists. CSP problems are known by some parameters such as density and tightness. Density of a CSP problem is defined as $p_1 = \frac{2m}{n*(n-1)}$ and its tightness is $p_2 = \frac{\#false\ assignments}{\#total\ assignments}$ in which $n$ is the number of variables and $m$ is the number of constraints.

### B. Distributed Constraint Optimization Problems (DCOP)

As mentioned, the goal of solving a CSP problem is to find out if the constraints can be satisfied. Similarly, a Constraint Optimization Problem (COP) problem can be defined based on the cost or gain function among variables. Following definition describes a distributed COP(DCOP) problems $\mathbf{P} = \langle \mathbf{V}, \mathbf{A}, \mathbf{D}, \mathbf{C} \rangle$ as (Yokoo & Durfee, 1991):

- A set of $n$ variables: $\mathbf{V} = \{v_1, \ldots, v_n\}$.
- A set of $g$ agents: $\mathbf{A} = \{a_1, \ldots, a_g\}$.
- Discrete, finite domains for each variable: $\mathbf{D} = \{\mathbf{D_1}, \ldots, \mathbf{D_n}\}$.
- A set of $m$ cost functions $\mathbf{C} = \{c_1, \ldots, c_m\}$, where each $c_h(d_{i,1}, \ldots, d_{i,j} | 1 \leq i, j \leq n)$ is a function where $c_h : D_{i,1} \times \cdots \times D_{i,j} \rightarrow \{q | q \in \mathbb{N} \ \wedge \ 0 \leq q \leq c_{max}\}$.

The set of $g$ agents are responsible for managing the variables of the problem. Each agent can be assigned one or more variables. That means an agent will manage the communication among its assigned variables with other variables and try to solve the whole problem while communicating with other agents. Moreover, $c_{max} \in C$ is the maximum possible cost (or gain) functions value. In this definition, it is assumed that the problem does not change over time, which means the constraints table is always the same. In other words, the graph structure of the problem never changes.

### C. Dynamic Distributed Constraint Optimization Problems (DynDCOP)

In the previous section static DCOP problems are introduced. In this section dynamic DCOP problems (DynDCOP) are explained. A DynDCOP problem can be defined as a sequence of static problems (Dechter & Dechter, 1988). Assume a DCOP problem $p_t$at time $t$ that changes to the problem $p_{t+1}$ at time $t+1$. The sequence of these static problems $\{P_0, P_1, \ldots, P_t\}$ over a period of time $(t)$ creates a dynamic DCOP problem. The difference between any two consecutive static problems $p_{t-1}$ and $p_t$ is the added $(\mathbf{c_t^a})$, removed $(\mathbf{c_t^r})$ or changed constraints. Therefore the problem $P_t$ can be written as (Verfaille & Jussien, 2005):

$$P_t = P_{t-1} + \mathbf{c_t^a} - \mathbf{c_t^r} \tag{1}$$

One can see that in the Eq. 1 the changed constraints are also included. A constraint change can be seen as removing it and adding it back with a new constraint table. As DynDCOP problems change over time, it is possible to measure their change rate (Mailler, 2005).

$$rate = \frac{dP}{dt} = \lim_{\Delta t \to 0} \frac{P_{t+\Delta t} - P_t}{\Delta t} \tag{2}$$

Eq. 2 can be expanded by including the definition of the problem $P_t$ at time $t$. (Mailler, 2005):

$$rate = \frac{1}{\Delta t} \sum_{i=t}^{t+\Delta t} \frac{|c_i^a| + |c_i^r|}{2} \tag{3}$$

where $\Delta t$ is the time that the problem $P_t$ changes to the problem $P_{t+\Delta t}$.

Since formulating the DynDCOP problem, several researchers have studied these problems' behaviors and have tried to model their behaviors. Some outstanding work on this area can be found in the work of (Ridgway & Mailler, 2015; Mailler & Zheng, 2014). They showed that DynDCOP problems follow the thermodynamic laws; therefore, the change of solution quality (energy) can be modeled and predicted. Some recent work in this area can be found in (Ridgway & Mailler, 2015; Samadidana & Mailler, 2019).

## III. Distributed Protocols

Various distributed algorithms have been designed to solve DCSP problems. In this study, Distributed Stochastic Algorithm (DSA) (Zhang et al., 2002), Distributed Breakout Algorithm (DBA) (Yokoo & K.Hirayama, 1996), and (Maheswaran et al., 2004) are selected to test and analyze the instability of the network. In the following subsections, each algorithm is described briefly. It must be noted that all of these algorithms have been adjusted to be able to solve DynDCOP problems.

### A. Distributed Stochastic Algorithm (DSA)

The first algorithm explained is Distributed Stochastic Algorithm (DSA) (Zhang et al., 2002). DSA is a simple algorithm that applies a hill-climbing approach to solve distributed problems. In this algorithm, agents who manage variables communicate and exchange information to get the latest information. The change is done by a fixed probability value $p$ whenever they have an improvement in their local solution. The local problem in DSA is finding the best assignments to their variables such that the constraints are satisfied (DCSP problems) or the cost(/gain) functions(DCOP problem) is minimized(/maximized). Each agent communicates directly with their direct neighbors using a message called *OK?* message.

```
procedure main
  while (not terminated) do
    update agent_view with incoming ok? (x_j, d_j)
messages;
    new_value ← choose_value;
    if new_value ≠ d_i do
      d_i ← new_value;
      send (ok?, (x_i, d_i)) to all x_j ∈ neighbors;
    end if;
  end do;
end main;
procedure choose_value
  if  d_i has no conflicts do
    return d_i;
  v ← the value with the least conflict (v ≠ d_i);
  if  v has the same or fewer conflicts than d_i
      and random < p do
    return v;
  else
    return d_i;
end choose_value;
```

Fig. 1. The procedures of the DSA-B algorithm

```
when received (ok, x_j, d_j) do
  if mode == wait_improve
    add message to queue;
    return;
  else
    add (x_j, d_j) to  agent_view;
    when received ok? messages from all neighbors do
      send_improve;
      mode ← wait_improve;
    end do;
  end if;
end do;
procedure send_improve
  current_eval ← evaluation value of current_value;
  improve_i ← possible maximum improvement;
  new_value ← the value which yields the best
improvement;
  send (improve, x_i, improve_i, current_eval) to
neighbors;
end send_improve;
```

Fig. 2. The procedures of the *wait_ok?* mode in DBA

In each time cycle, agents calculate their improvement values based on their local value and the received messages from their direct neighbors. Each *OK?* message contains the current value assigned to the sender's variable. Based on the probability $p$, if agents find an improvement, they change their values and communicates the new value to their neighbors. This process repeats until a fixed number of cycles are reached, or an optimal solution is found. It is worth mentioning that DSA does not guarantee to find the global optimal solution (Leite, Enembreck, & Barthes, 2014). Despite this issue, DSA is very fast and in many problems yields a good solution. From five variants of DSA, DSA-B is used in this paper that allows an agent to change their values despite not having an improvement(Fig. 1).

### B. Distributed Breakout Algorithm (DBA)

The Distributed Breakout Algorithm (DBA) (Yokoo & K.Hirayama, 1996) is another distributed algorithm similar to DSA works based on communication. However, DBA uses two kinds of messages called *ok?* and *improve?*. The *ok?* message contains the variable's current value, and the *improve?* message includes the amount of improvement of the variable's local problem's solution. After an agent receives the *ok?* messages from its neighbors, it calculates its improvement and communicates it with its direct neighbors through *improve?* messages. Once an agent receives all the *improve?* messages, it compares its improvement with its neighbors, and if it has a better improvement, it will change its value. DBA uses a weighting mechanism to escape from the quasi-local-minimum by changing the weight of violated constraints.

Figures 2 and 3 show the various procedures of DBA.

### C. Max-Gain Message Algorithm

Max-Gain Message Algorithm (MGM) is another distributed algorithm that, similar to DBA, is a two-phase protocol (Maheswaran et al., 2004). Each agent calculates its maximum gain value and propagates it through the network. If an agent has a higher gain value, it changes its value and notifies its neighbors. Nonetheless, MGM also can suffer from the local minimum issue. Brief pseudocode of MGM is presented in Fig. 4.

## IV. Test Results

As mentioned in Section II, several studies have confirmed the negative effect of information stagnancy on distributed problem-solving. Nevertheless, more work is needed to gain a comprehensive view of the effect of information stagnancy. This work provides a detailed examination of various information stagnancy sources, including message loss, delay, and distortion. In order to investigate the effect of communication issues comprehensively, both static and dynamic DCOP problems are evaluated. Moreover, various test parameters are chosen to demonstrate the effects of information stagnancy better. Each test scenario is run 50 times, and the average function is used to summarize the result. A test scenario includes a DCOP problem with a determined tightness, density, and the number of variables. Furthermore, various amounts of noise and loss probability are used to have a combination of

```
when received (improve, x_j, improve_j, eval) do
  if mode == wait_ok
    add message to queue;
    return;
  else
    record message;
    when received improve messages from all neighbors
do
      send_ok;
      clear agent_view;
      mode ← wait_ok;
    end do;
  end if;
end do;
procedure send_ok
  if improve_i is better than all of my neighbors
    current_value ← new_value;
  end if;
  when in a quasi-local-minimum do
    increase the weights on all violated constraints;
  end do;
  send (ok?, x_i, current_value) to neighbors;
end send_ok;
```

Fig. 3. The procedures of the *wait_improve* mode in DBA

```
procedure main
  initialize;
  while (not terminated) do
    send current value to neighbors;
    get all neighbors value;
    calculate and send_maximum gain mg;
    collect all neighbors maximum gain_ok;
    if mg > all neighbors maximum gain_ok;
      set new value to value that gives mg;
    end if;
    send_improve;
  end do;
end main;
```

Fig. 4. The Max-Gain Message algorithm

communication degradation. The complete list of test scenario parameters is described in the following sections. Usually, the gain/cost value is used to evaluate the performance of DCOP problems.

As in DCOP problems, agents communicate the most up-to-date information on a cycle-based time; the solution at the end of the last cycle is used. In other words, this solution is the one that the solver converges to after a certain period. As each distributed algorithm completes the task of solving a single instance of the problem at a different time, another parameter called cycle is used. This parameter allows a distributed algorithm to run for a determined number of times it takes to receive/send the messages from/or the agents. For example, in DSA, a cycle for an agent is the amount of time required to receive the $ok?$ messages and decide if they have a better value and sending the new information. In DBA, an agent either processes $ok?$ or $improve?$ messages in each cycle. Therefore, DBA is a two-phase algorithm that needs cycles to complete the solving process. Similarly, MGM works in two phases to complete solving the problem. It is worth mentioning that this process repeats until either a certain number of cycles are reached, or the algorithms reach an expected solution. In this study, each algorithm runs 1000 cycles.

It is worth mentioning that it is assumed that the algorithms do not use a checking system to verify the correctness of messages. This assumption is necessary because having a verification mechanism reduces the noise issue with messaging loss problem (Samadidana & Mailler, 2017) however, having a verification mechanism is not always possible due to various reasons such as cost, power, or bandwidth issues; therefore, the noise problem is not always avoidable. For example, adding a checksum to the messages might increase the processing time in addition to the size of the messages.

Two subsections continue this section. The first one provides the results and analysis for the static DCOP problems, and the second one investigates the dynamic ones.

### A. Static DCOP Problems

The first part of the tests is dedicated to static problems. In the static problems, the problems do not change over time; hence the distributed algorithms will solve the problems over a fixed number of cycles. Then the final solution (convergence point), which is obtained at the end of the last cycle of solving the problem, is used to determine the algorithms' performance. The parameters used in this section are as following:

- number of variable = 100
- number of domain value = 3
- density = $\{0.02, 0.035, 0.055\}$
- tightness = 0.33
- min cost = 1
- max cost = 10
- noise probability =$\{0.1, 0.2, 0.3\}$

For each combination of parameters, 50 test runs were run, and the average function is used to analyze the result. Each problem contains 100 variables (agents), and each variable can be assigned three domain values. The minimum cost of each constraint is set to 1 and the maximum cost to 10. In addition, to evaluate the impact of the problem densities, three densities are chosen. In each test scenario, various noise probabilities are chosen to investigate the amount of impact.

*1) Noisy Environment (No Messages Loss):* Let us begin with the scenario where all messages are delivered with a corruption chance. Tables I, II, and III present the total costs (convergence points) of the solutions for DSA, DBA, and MGM respectively. It is clear that the high chance of noise

has decreased the solutions' quality for all the algorithms. Especially, the noise chance of 0.3 shows a more negative impact on the result. The outputs suggest that DSA performs worse than DBA and MGM. This completely makes sense as DBA and MGM send the latest information constantly. Therefore, even if the noise issue corrupts some messages, in the next cycles, the agents have the chance to send the correct information. In DSA, not all the agents send the latest information in each cycle; therefore, message corruption impacts the solution more dramatically. Even if an agent gets a chance to communicate its value after some cycles, the noise chance might impact it. Besides, the results suggest that MGM works slightly better than DBA, although the difference becomes less apparent as the noise chance increases. Fig. 5 provides a summary of these three algorithms' results for various noise chances and the density $p = 0.035$.

The result also shows an interesting behavior of MGM and DBA. When both algorithms are tested in an environment with a chance of noise 0.1, they yielded better results than the problems where the communication is perfect. The reason for this peculiar phenomenon is that both DBA and MGM may get stuck in local minima; however, the small amount of noise helped them to be able to improve the solution.

It can be concluded that algorithms that communicate the information more often have a clear advantage in static problems when the noise exists. It must be noted that this advantage comes at the cost of more communication. The number of messages exchanged in DBA and MGM is much higher than the DSA's. In networks with limited bandwidth, this massive number of communication can be problematic. Therefore, a justification needs to be made to choose the proper algorithm.

TABLE I
CONVERGENCE POINTS OF DSA FOR DENSITIES $p = \{0.02, 0.035, 0.055\}$ (SAMADIDANA, 2021)

| Noise | Density | | |
|---|---|---|---|
| | ***0.02*** | ***0.035*** | ***0.055*** |
| ***No Noise*** (Samadidana, 2019) | 229.91 | 474.63 | 848.61 |
| ***0.1*** | 236.23 | 489.31 | 864.43 |
| ***0.2*** | 249.10 | 504.32 | 889.27 |
| ***0.3*** | 255.97 | 513.19 | 910.51 |

TABLE II
CONVERGENCE POINTS OF DBA FOR DENSITIES $p = \{0.02, 0.035, 0.055\}$ (SAMADIDANA, 2021)

| Noise | Density | | |
|---|---|---|---|
| | ***0.02*** | ***0.035*** | ***0.055*** |
| ***No Noise*** (Samadidana, 2019) | 223.59 | 472.71 | 846.16 |
| ***0.1*** | 215.06 | 455.21 | 819.17 |
| ***0.2*** | 235.42 | 485.11 | 845.40 |
| ***0.3*** | 257.57 | 507.17 | 872.94 |

*2) Noisy Environment (With Message Loss):* In this part, message loss is included in the parameters. Loss probabilities that are tested in this study are 0.1, 0.2, and 0.3. Tables 6, 7, and 8 show the results of solving static problems which are impacted by both noise and message loss. When compared with the results of solving static problems in a perfect environment (no noise and loss issue), it can be observed that all of the algorithms have been impacted dramatically by the communication issues. With a lower chance of message loss and noise, the difference among DSA, MGM, and DBA results is small; however, with a higher chance of communication issues, the difference becomes more evident. For example, in Fig. 8, when the noise and message loss chances are both equal to 0.3, DBA and MGM work much worse than DSA. These results can be compared to the previous sections' results in which DBA and MGM were showing better results. To understand the reason for this difference, the structure of DSA, DBA, and MGM must be reviewed. As mentioned in DSA, agents send their information when they have an improvement based on a fixed probability. When the DCOP problem is impacted by noisy communication, the message loss can prevent agents from getting the wrong information. It is worth mentioning that in DSA-B, agents are allowed to change their value even if they do not have an improvement. However, in DBA and MGM, the issue of noise is doubled by the message loss problem. Both DBA and MGM need information from all of their neighbors in order to decide about their values. When some messages are lost, the decision-making process is delayed. This problem is worsened when the noise issue causes delivering invalid messages.

TABLE III
CONVERGENCE POINTS OF MGM FOR DENSITIES $p = \{0.02, 0.035, 0.055\}$ (SAMADIDANA, 2021)

| Noise | Density | | |
|---|---|---|---|
| | ***0.02*** | ***0.035*** | ***0.055*** |
| ***No Noise*** | 223.47 | 474.47 | 850.28 |
| ***0.1*** | 217.18 | 449.6 | 806.49 |
| ***0.2*** | 236.28 | 482.57 | 843.49 |
| ***0.3*** | 253.78 | 506.64 | 865.9 |

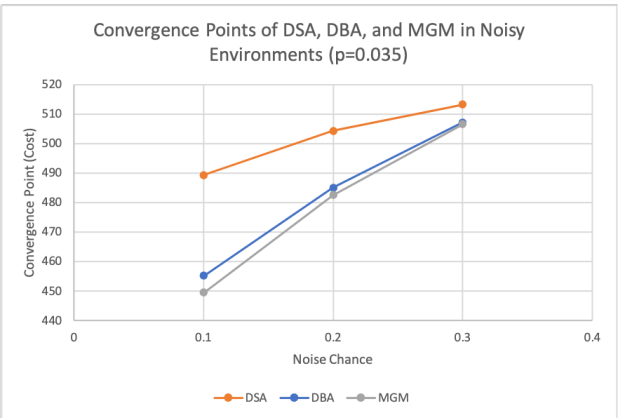


Fig. 5. Comparison of convergence points of DSA, DBA, and MGM in noisy static DCOP problems with density $p = 0.035$ (Samadidana, 2021)

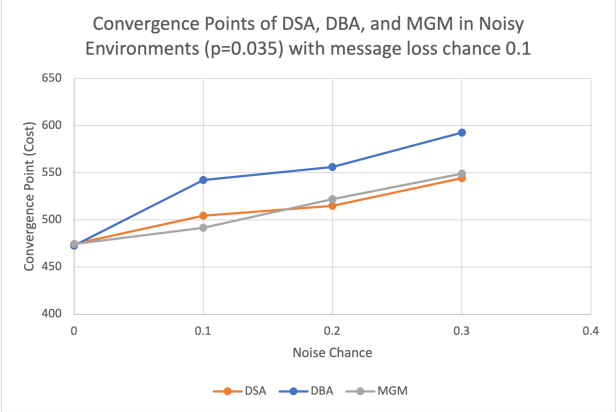


Fig. 6. Comparison of convergence points of DSA, DBA, and MGM in noisy static DCOP problems with density $p = 0.035$ and loss chance $0.1$

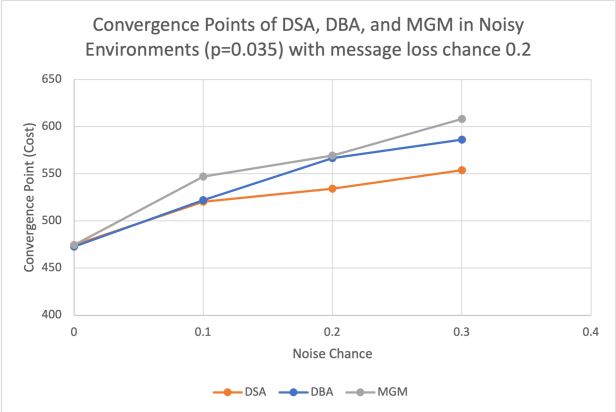


Fig. 7. Comparison of convergence points of DSA, DBA, and MGM in noisy static DCOP problems with density $p = 0.035$ and loss chance $0.2$

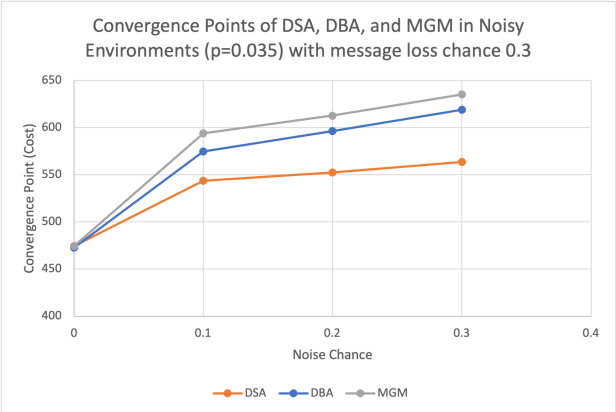


Fig. 8. Comparison of convergence points of DSA, DBA, and MGM in noisy static DCOP problems with density $p = 0.035$ and loss chance $0.3$

### *B. Dynamic DCOP Problems*

In order to have a comprehensive evaluation of message loss and noise, Dynamic DCOP problems (DynDCOP) are also evaluated. Similar to the previous sections, first, the noise issue in dynamic problems is tested. Then, the effect of having both loss and noise issues on a variety of problems has been tested. The parameters used in this section are the ones used in the static DCOP problems with the addition of the change in the problems.

- all static problems parameters
- change rate=$\{1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90\}$ of the messages

Here the change rate means the number of added or removed constraints. For example, a change rate of $1$ means one constraint is removed, and a new constraint is added. It is evident that in this case, the density of the problems does not change. Each test scenario is tested 50 times, and similar to the static problems, the average function is used to explain the result. First, the DynDCOP problems are tested under noisy conditions and without the message loss issue, followed by the results of solving DynDCOP problems with both noise and message loss issues.

*1) Noisy Environment (No Messages Loss):* As mentioned, dynamic DCOP problems change over time. In this study, it is assumed that the change rate is constant. Hence, in each cycle, only a determined number of constraints will be changed. Also, the change probability for each constraint is the same. Figures 9, 10, and 11 provide an overview of the results for density $p = 0.035$. The results confirm that by increasing the change rate of the problems, they converge to a worse result. Besides, in the absence of noise, algorithms work better (Samadidana, 2019). This is not surprising as distributed algorithms depend on communications; therefore, corruption in messages leads to lower quality solutions. Furthermore, as the noise chance increases, the results become worse; however, the difference among the results for different noise chances becomes smaller at the higher change rates. This phenomenon suggests that the high change rate overshadows the noise issue. Clearly, at the high change rates, the algorithms cannot keep up with the changes. Therefore, agents cannot make correct decisions. Even sending the correct result might not work as there is a chance that the constraints have been changed; therefore, the information will be out of date.

As shown in Fig. 9, DSA is resistant to noise in dynamic DCOP problems, especially for a lower chance of noise such as $0.1$ the results are very close to the instances with perfect communication. One reason for the results can be the quickness of DSA to adapt itself to the changes in the environment. Therefore, even with some noise, it still can converge to a good solution. Nevertheless, at the very high change rates, DSA converges to the point (equilibrium point) where no progress can be made. More information can be found in the work of (Ridgway & Mailler, 2015)

Fig. 10 shows the results for DBA. Similar to DSA, with the increase in the noise probability, the results become worse.

However, in DBA, the difference among the results of solving the problems when the communication is perfect is higher than DSA's. As DBA works in a two-phase model, its decision-making process takes a longer time. Therefore, while the agents are processing the data for the current instance of the problem, the problems have changed to a new instance. The noise issue worsens the situation as the delivered messages for the unchanged constraints might be invalid.

MGM's result is very similar to the DBA's, although it shows a slightly better performance. The propagation of data in MGM by the agents can reduce the error of decision-making. However, it works in two phases that can create an unwanted delay in dynamic programs. That is why both DBA and MGM show a worse result compared to DSA.

a summary of solving DynDCOP problems with noisy communication is presented in the figures 12, 13, 14. As discussed, DSA outperforms both DBA and MGM while both DBA and MGM produce very similar results.

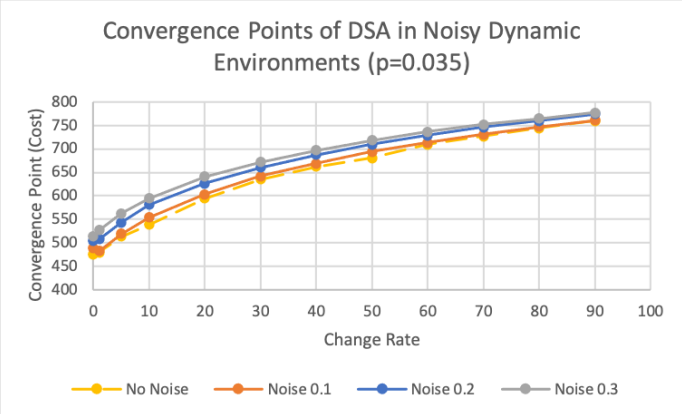


Fig. 9. Comparison of convergence points of DSA in a noisy dynamic environment with problem density $p = 0.035$ (Samadidana, 2021)

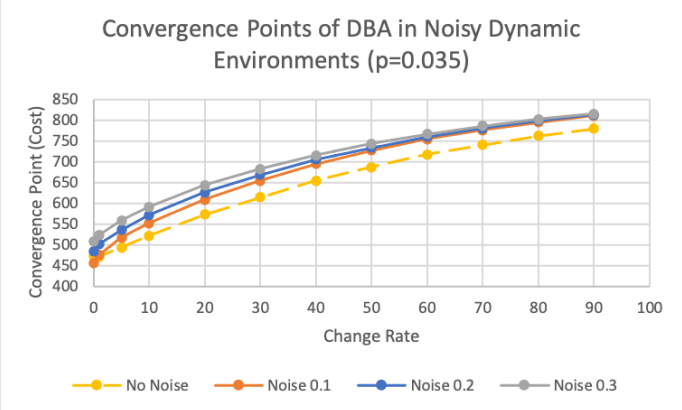


Fig. 10. Comparison of convergence points of DBA in a noisy dynamic environment with density $p = 0.035$ (Samadidana, 2021)

*2) Noisy Environment (With Message Loss):* The last part of the tests is dedicated to solving DynDCOP problems while the problems change over time, and the communication is

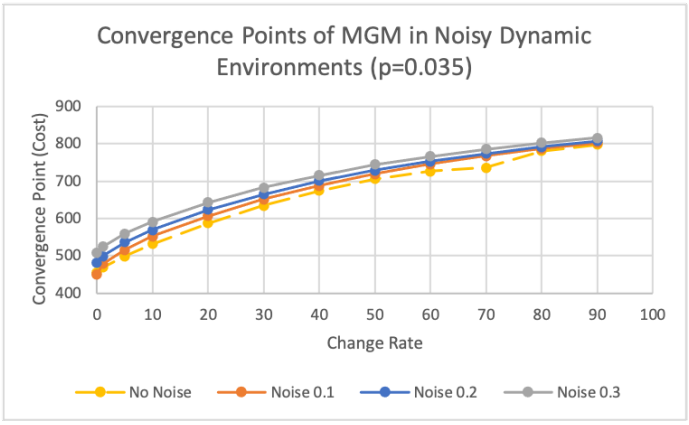


Fig. 11. Comparison of convergence points of MGM in a noisy dynamic environment with density $p = 0.035$ (Samadidana, 2021)

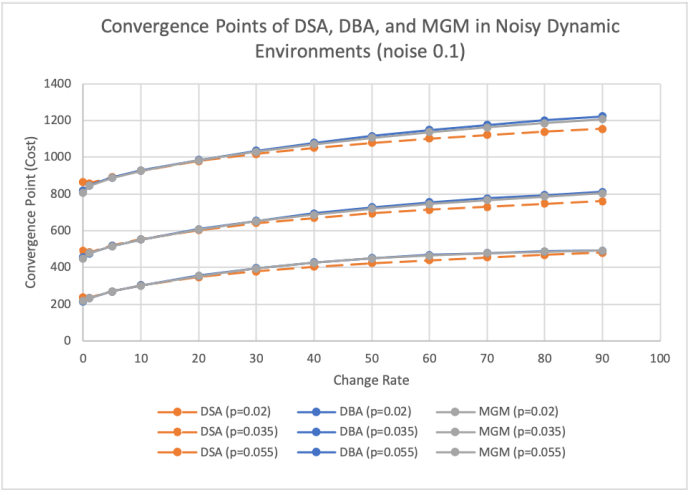


Fig. 12. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.1 (Samadidana, 2021)

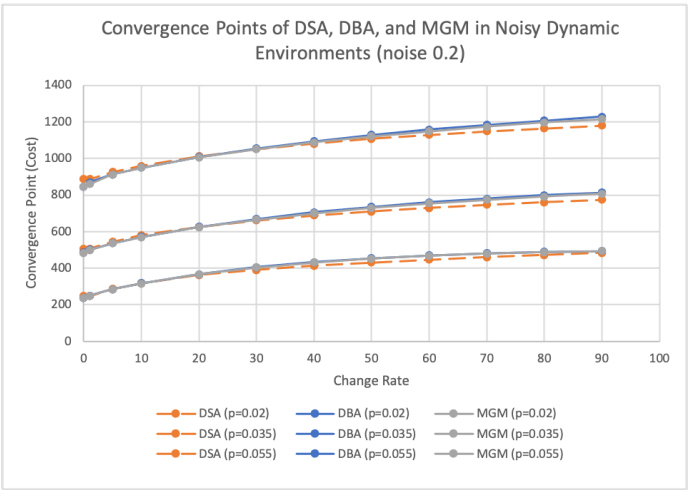


Fig. 13. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.2 (Samadidana, 2021)

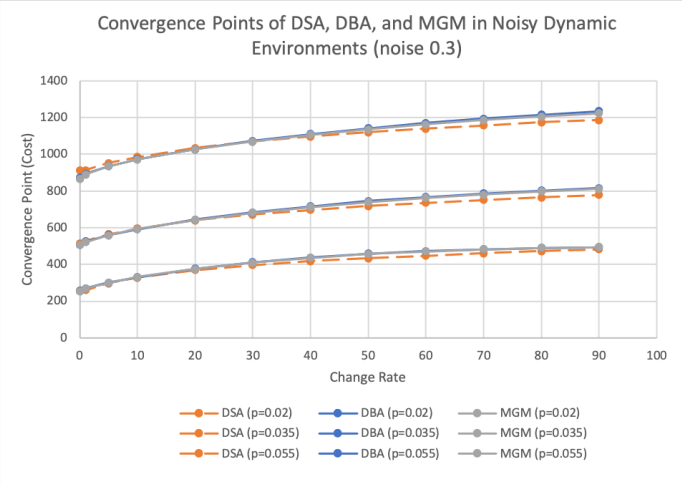


Fig. 14. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.3 (Samadidana, 2021)

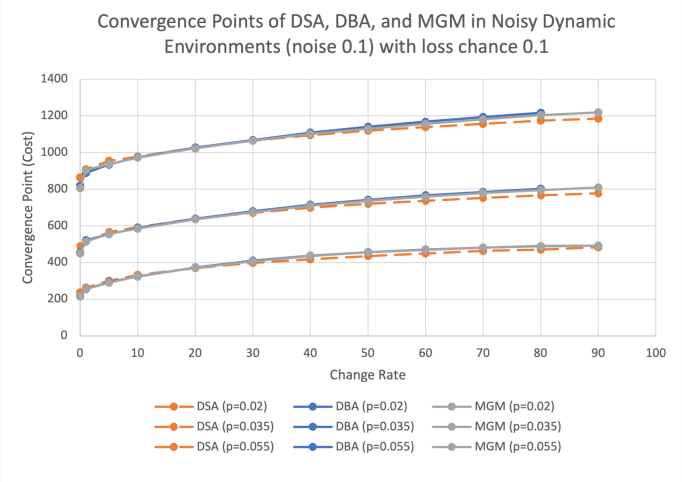


Fig. 15. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.1 and loss chance of 0.1

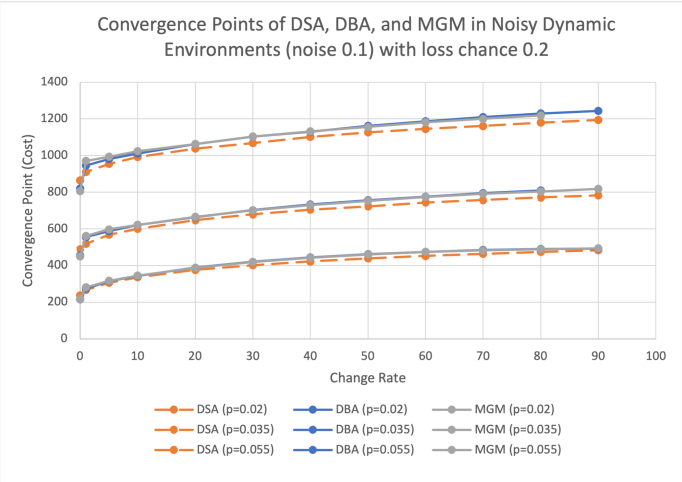


Fig. 16. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.1 and loss chance of 0.2

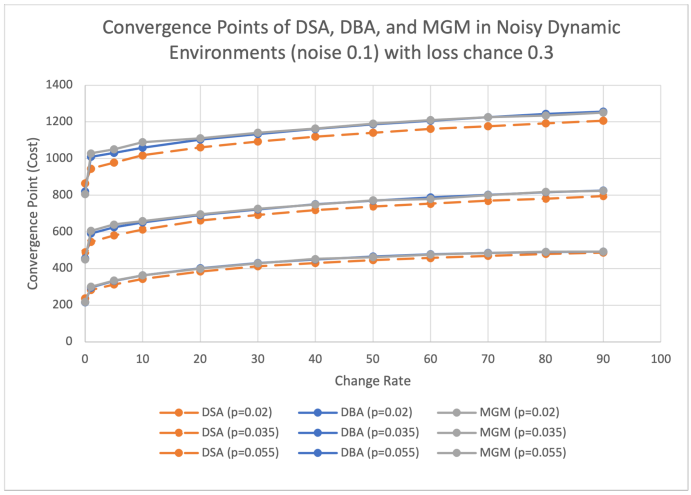


Fig. 17. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.1 and loss chance of 0.3

impacted by both noise and message loss. The parameters used in this section are the ones used in the previous sections in addition to the message loss probability of 0.1, 0.2, and 0.3. All three distributed algorithms have been tested under the mentioned conditions to explore the effect of both noise and message loss.

Figures 15-23 show the result for DynDCOP problems with various densities $p = 0.02, 0.035, 0.055$ for DSA, DBA, and MGM respectively. Similar to the DynDCOP problems with noise issue and without the message loss problem, DSA had provided better results. Here, DSA outperforms two other algorithms. MGM and DBA had very similar results. Interestingly in the problems with higher density, the difference among the results becomes less apparent. This suggests that algorithms cannot keep up with the number of changes in the problems in the more dense problems. It must be noted that with a higher density amount, the number of variables that are impacted increases. Hence, the error amount produced by noise or message loss also increases.

All algorithms show a sharp increase in the convergence points than the static problems without noise or message loss. For example, for a problem with a density of 0.02 DSA, DBA, and MGM yield 236.23, 215.06, and 217.18, however, with a change rate of 1 and noise and message loss probabilities both equal to 0.1, the convergence pints become 264.32, 254.94, and 256.65 respectively. With the same chance of noise and noise probability 0.2, the convergence points increase to 267.15, 273.92, and 280.81. Similarly, for all other densities, noise and message loss probabilities pattern repeats. By looking at another side of the results when the change rate is very high, it is visible that none of the algorithms can improve the solutions. Even DSA that works in one phase and is fast cannot produce better results compared to DBA and MGM.

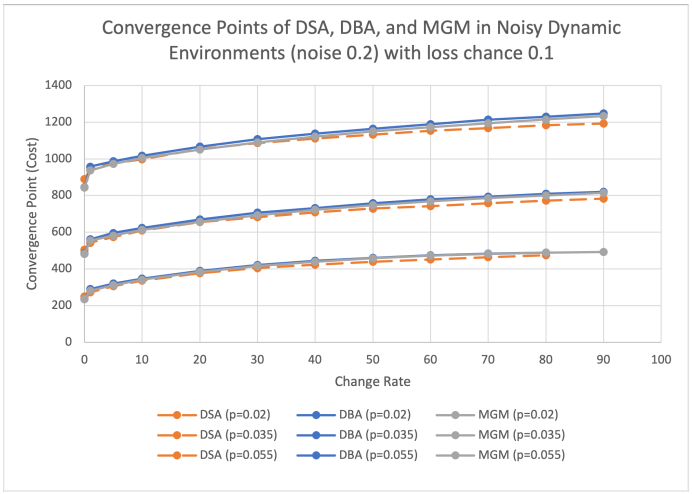


Fig. 18. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.2 and loss chance of 0.1

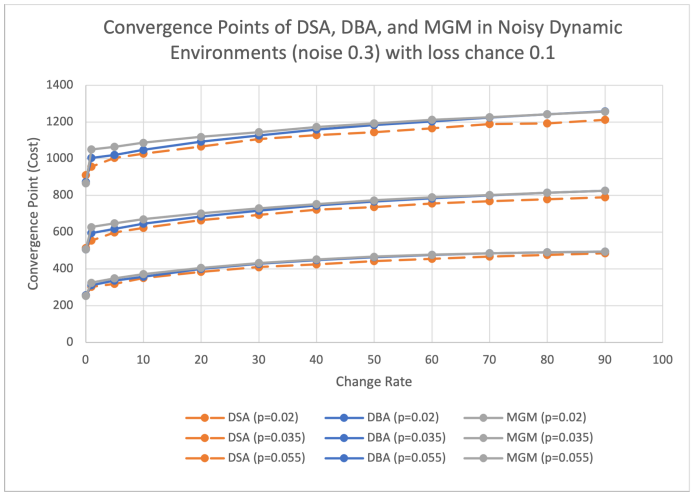


Fig. 21. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.3 and loss chance of 0.1

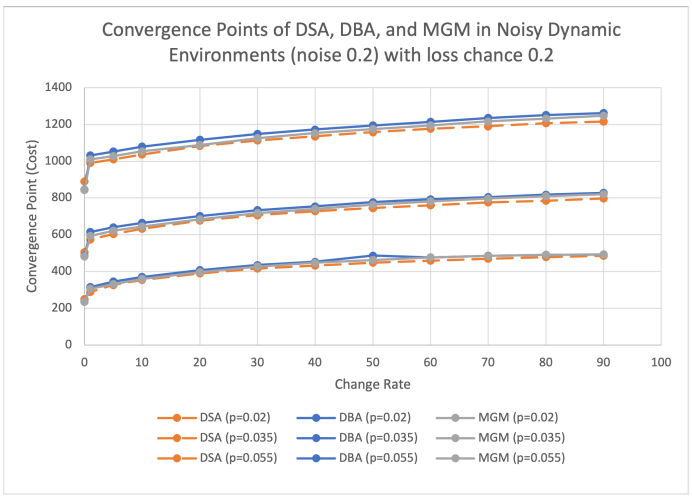


Fig. 19. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.2 and loss chance of 0.2

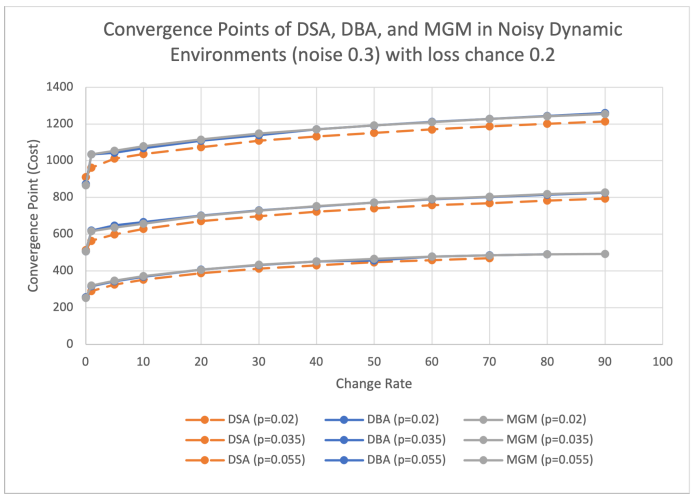


Fig. 22. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.3 and loss chance of 0.2

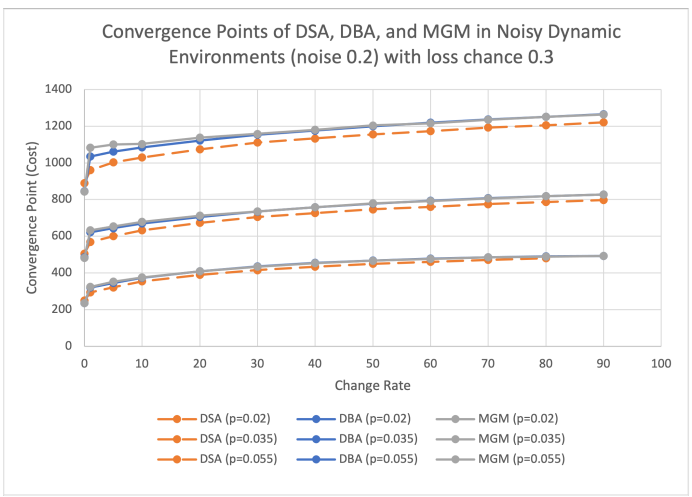


Fig. 20. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.2 and loss chance of 0.3

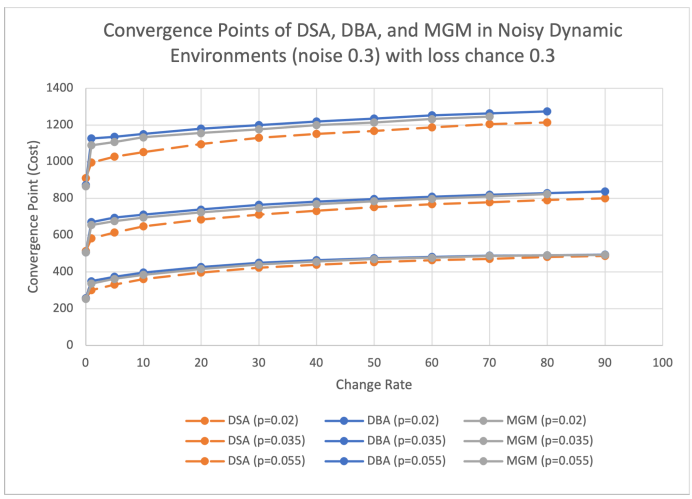


Fig. 23. Comparison of convergence points of DSA, DBA, and MGM in noisy dynamic environment with noise chance of 0.3 and loss chance of 0.3

## V. Conclusion

The advances in network technologies have provided a robust communication environment for many distributed systems. Nevertheless, despite all the progress, not all distributed systems work in a perfect environment. Issues such as noise and message loss may dramatically impact the performance of distributed algorithms. In this study, three distributed algorithms DSA, DBA, and MGM, are tested to analyze the effect of both noise and message loss. The result confirms that all of the algorithms are highly impacted by communication instability. DSA shows better results in many cases; however, in dynamic DCOP problems, its performance is degraded. Results confirm that fast, responsive algorithms are better candidates for solving DynDCOP problems. In future works, the DynDCOP problems in which other problems' parameters such as density change over time will be studied.

## Acknowledgment

The author would like to thank the anonymous reviewers for their time.

## References

Alemany, M., Esteso, A., Ángel Ortiz, & del Pino, M. (2021). Centralized and distributed optimization models for the multi-farmer crop planning problem under uncertainty: Application to a fresh tomato argentinean supply chain case study. *Computers & Industrial Engineering*, *153*, 107048.

Andrews, J. G., Buzzi, S., Choi, W., Hanly, S. V., Lozano, A., Soong, A. C. K., & Zhang, J. C. (2014). What will 5g be? *IEEE Journal on Selected Areas in Communications*, *32*(6), 1065-1082. doi: 10.1109/JSAC.2014.2328098

Dechter, R., & Dechter, A. (1988). Belief maintenance in dynamic constraint networks. In *Proceedings of the 6th national conference on artificial intelligence (aaai-88)* (p. 37-42).

Kiran, D. (2019). *Chapter 35 - internet of things,production planning and control*. Butterworth-Heinemann.

Leite, A. R., Enembreck, F., & Barthes, J.-P. A. (2014). Distributed constraint optimization problems: Review and perspectives. *Expert Systems with Applications*, 5139–5157.

Maheswaran, R. T., Pearce, J. P., & D, M. T. (2004, September). Distributed algorithms for dcop: A graphical-game-based approach. *In Proc. Parallel and Distributed Computing Systems PDCS*, 432–439.

Mailler, R. (2005). Comparing two approaches to dynamic, distributed constraint satisfaction. In *Proceedings of the fourth international joint conference on autonomous agents and multiagent systems (aamas)* (p. 1049-1056).

Mailler, R., & Zheng, H. (2014). A new analysis method for dynamic, distributed constraint satisfaction. In *Proceedings of the 2014 international joint conference on autonomous agents and multiagent systems (aamas)* (Vol. 3, pp. 901–908).

Mira, I. A. . F. (2019). Iot security threats analysis based on components, layers and devices. *American Journal of Science and Engineering*, 1–10.

Ridgway, A., & Mailler, R. (2015). Dynamic theoretical analysis of the distributed stochastic and distributed breakout algorithms. In *Proceedings of the 14th international conference on autonomous agents and multiagent system.*

Ridgway, A., & Mailler, R. (2015). On predictions for dynamic, self-adaptive techniques in dyndcsp's. In *2015 ieee/wic/acm international conference on web intelligence and intelligent agent technology (wi-iat)* (Vol. 2, p. 265-272). doi: 10.1109/WI-IAT.2015.156

Russell, S. J., & Norvig, P. (2020). *Artificial intelligence: A modern approach, 4th edition*. Prentice Hall.

Samadidana, S. (2019). The effects of information stagnancy on distributed problem solving. *ProQuest Dissertations and Theses*, 166.

Samadidana, S. (2021). Exploring the effect of noisy environment on distributed problem solving. In *2021 ieee 11th annual computing and communication workshop and conference (ccwc)* (p. 0342-0348).

Samadidana, S., & Mailler, R. (2017). Solving dcsp problems in highly degraded communication environments. In *Proceedings of international conference on web intelligence (wi17)* (p. 348—355).

Samadidana, S., & Mailler, R. (2019). Using temporal awareness to improve distributed problem solving. In *Ieee 9th annual computing and communication workshop and conference (ccwc)* (pp. 342–348). Retrieved from doi:10.1109/CCWC.2019.8666568

Samadidana, S., Paydar, M., & Jouzdani, J. (2017). A simulated annealing solution method for robust school bus routing. *International Journal of Operational Research*, *28*, 307–326. doi: https://doi.org/10.1504/IJOR.2017.081908

Verfaille, G., & Jussien, N. (2005). Constraint solving in uncertain and dynamic environments: A survey. *Constraints*, *10*(3), 253-281.

Wahbi, M., & Brown, K. N. (2014). The impact of wireless communication on distributed constraint satisfaction. In *Proceedings of the 20th international conference on principles and practice of constraint programming* (pp. 738–754). Lyon, France.

Yao, W., Qi, N., Wan, N., & Liu, Y. (2019). An iterative strategy for task assignment and path planning of distributed multiple unmanned aerial vehicles. *Aerospace Science and Technology*, *86*, 455-464.

Yokoo, M., & Durfee, E. H. (1991). *Distributed constraint optimization as a formal model of partially adversarial cooperation* (Tech. Rep. No. CSE-TR-101-91). Ann Arbor, MI 48109: University of Michigan.

Yokoo, M., & K.Hirayama. (1996). Distributed breakout algorithm for solving distributed constraint satisfaction problems. *International Conference on Multi-Agent Systems(ICMAS)*, 401–408.

Zhang, W., Wang, G., & L.Wittenburg. (2002). Distributed stochastic search for constraint satisfaction and optimization:parallelism, phase transitions and performance. *In proceedings of the AAAI workshop on probabilistic Approaches in Search*, 53–59.