Multi-Agent Systems
Project Work

Marcell Zoltán Szalontay

August 27, 2017

Contents

1 Introduction

Multi-agent system nowadays a highly researched field that has proven its utilities in many areas. Decisive advantage that these systems with collective intelligence can be used to solve problems that are difficult or impossible for an individual agent. One of most important use of multi-agent systems has been closely related to the distributed AGV problem. With this area, we can give a manifest solution to the following problem: i) task and resources allocation ii) communication protocols.
The aim of the project is to develop a DARP problem solving system, where the task allocation and the communication protocol will be a key. During the task, two types of communication protocols will be used: i) control net protocol - CNet, and ii) dynamic control net protocol - DynCNet.

1.1 Objectives

In this project, I examine a dial a ride problem (DARP), that belongs to the family of pickup and delivery problem (PDP). During the study I simulate a taxi company's operation where I used decentralized control through various communication protocols: i) the contract net protocol which also known as CNet protocol, and ii) the dynamic version of it the DynCNet. I also use some additional constraints, such as battery supplies problems and limited communication range between the participants. During the task allocation, agents will be each other's rivals who are trying to make the most advantageous offer to customers so that they can deliver them. Nevertheless, as an organization, the behavior of agents can be called co-operation, because as the closest agent always carries the customer, the system seeks optimal optimization. This kind of cooperative task execution also known as task-sharing.
The purpose of this project to investigate the efficiency and the performance of CNet and DynCNet protocol and to show how the system response time depends on the number of agents and the deadline value with CNet and DynCNet.

The objectives of the project are summarized as follows:

- Implement the communication protocols such as: CNet and DynCNet
- Investigate the efficiency and the performance of these communication protocols
- To show how the system response time depends on the number of agents and the deadline value with the different communication protocols

1.2 Hypotheses

Based on the above objectives I have formulated the following hypotheses for the task:

1. The DynCNet gives a better utilization for the vehicles than CNet protocol.
2. We can have more finished transport using DynCNet than using CNet.
3. DynCnet always generates more messages regardless of time and the number of agents compared to CNet.
4. Both algorithms increase efficiency by increasing the number of taxi agents.
5. Both algorithms increase the utilization of vehicles by increasing the number of customers.

1.3 Variables

Each variables can be different depending on their effect on the system. At this stage I will list the variables used in the experiments by the following categories:

- The group of variables that are systematically changed during the experiment are called **independent variables**.
- The variables to which the independent variables will affect will be called **dependent variables**.
- The variables that remain the same throughout the experiment are called **other variables**.

The variables used during the experiment are classified into these three groups:

- **Independent variables:**

1. number of taxi agents	value type: integer	value: 20-40
2. communication protocol	value type: algorithm	value: CNet, DynCNet
3. deadline	value type: time(min)	value: 1-15

- **Dependent variables:**

1. messages	value type: integer
2. utilization	value type: percent
3. avg waiting time	value type: time (tick)
4. finished transports	value type: integer

- **Other variables:**

1. Taxi transport start cost	value type: integer	value: 4
2. Taxi transport cost/km	value type: integer	value: 2
3. Battery	value type: integer	value: 6000
4. Taxi capacity	value type: integer	value: 3
5. Number of Taxi Bases	value type: integer	value: 8

2 Theory

The following system - like most MAS environments - implements a dynamic environment where new customers may be present during the action, and agent status changes also affect how others respond. During communication, it is indispensable for the agents to:

1. ability to "physically" exchange information
2. common understanding
3. common language
4. interaction strategies / protocols

The most spreaded common language is the Agent Communication Languages (ACL), which support intentional communication, and its communication protocol independent. Communication between agents and customers is based on a communication protocol. Two different algorithms have been implemented on the system:

- contract net protocol - CNet
- dynamic contract net protocol - DynCNet

In the following section, I present the structure and operation of these two algorithms.

2.1 CNet protocol

The Contract Net Protocol itself is one of the most important paradigms that has been devolved to decentralized task allocation in distributed artificial intelligence. The significance of the protocol lies in the fact that this was the first breakthrough in to use a negotiation process involving a mutual selection by both managers and contractors [1]. In the other hand, the knowledge.sources need to work together, since none of them have enough information to solve the whole problem - the network reaches through the mutual interaction of the parties so that it can respond to the process as a result of the team's cooperation.

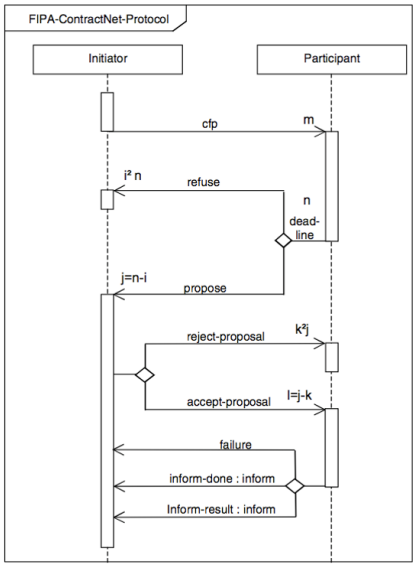

Figure 1: The state diagram of the CNet protocol.

All the devices are in communication are connected with the rest, so everyone can communicate with everyone else by sending and receiving messages. In the present project, it is important to

note that communication is from one aspect that some node can only communicate with certain means - those who can solve their problem. For example, passengers should be in contact with only taxi agents who are capable of transporting them from A to B.
Negotiation requires four basic principles:

1. it is a local process that does not involve centralized control
2. there is a two-way exchange of information
3. each party to the negotiation evaluates the information from its own perspective, and
4. final agreement is achieved by mutual selection.

The communication tools / agents in the network are called nodes, all of which make up the network. If a task is to be executed, a contract will be created between two nodes.Each node in the net takes on one of two roles related to the execution of an individual task: manager or contractor. A **manager** is responsible for monitoring the execution of a task and processing the results of its execution. A **contractor** is responsible for the actual execution of the task.
The contract itself is made between the two parties by local selection based on bidirectional information transfer. During the contract the contractor send a local broadcast message within their own communication range, to the free agents inside it issues and issue a request for bids. After the customer request, under a certain tolerance - pre-set message deadline - the bids arrived from the managers. Then the customer considers the bids and selects and accepts the most advantageous, and rejects the rest. This is where the agreement is reached between the best bidder manager and the contractor. The Contract Net protocol status diagram is shown in Figure 1.
This protocol provides a very effective solution to address this problem, and agents can perform their task with extreme utilization. However, we need to see that the problem and the environment in which agents work - as is the case with most real-life problems - is dynamic. As a result, it does not work with optimum results in certain problem situations. DynCNet can provide a solution for this problem as the dynamic version of CNet, which is described below.

2.2 DynCNet protocol

Previously, the CNet protocol was introduced and it was highlighted in its shortcomings. Below, I will present the dynamic version of the CNet protocol, DynCNet, which is an obvious solution to the problem that can not be solved by CNet, and these problems will be outlined.
The basic idea of the communication protocol is the same as the CNet build - so the protocol relies on different states and actions that agents perform while performing the task. During each task, every single agent's has a task to do, but none of them could solve the entire task [2] These parts what the agents do called states. DynCNet's operation - as it is significantly more complex than CNet - is illustrated in its statemachine in Figure 2.

In CNet, when a new transport agent appears with the local broadcast message, it communicates this to the AVGs that bid for it, and then the transport agent selects the most appropriate and ready agreement between the two parties. DynCNet makes this process more flexible.
DynCNet has the ability to allow both AVGs and transport agents to switch partners. In the case of CNet, the contract can no longer be dissolved if the two parties agree, then no change will take place. However, in DynCNet, if you have a better bid in the meantime - both for AVG and transport agent - may choose to withdraw from the previous agreement. That is why we call it a provision agreement at this stage of the task. The two parties may so decide, even though AVG did not actually reach the point of delivery of the transport agent and did not start shipping.

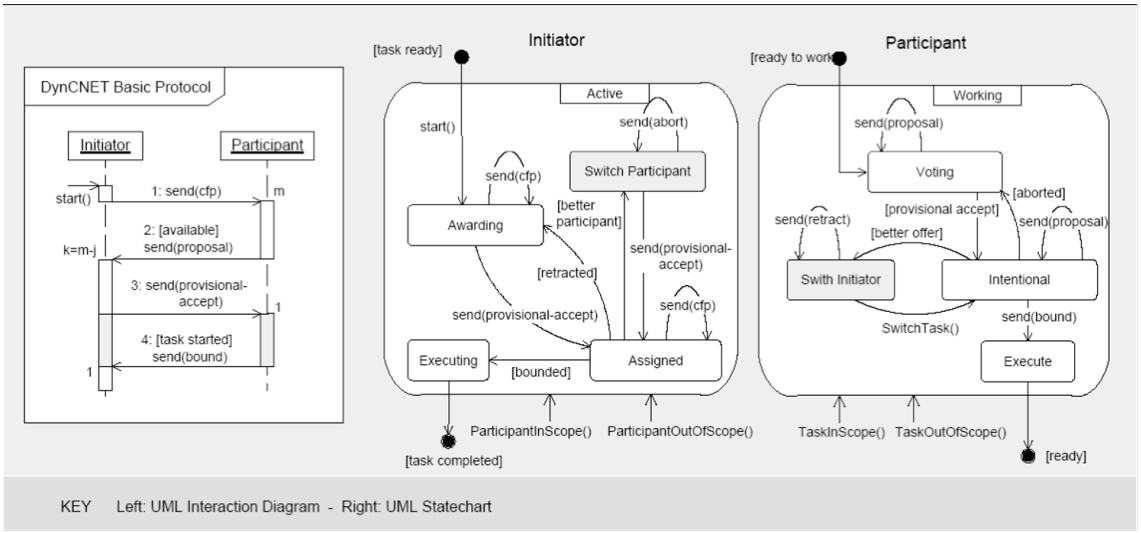

Figure 2: The statemachine of the DynCNet protocol (The DynCNet protocol in the left, The point of view of an initiator in the middle and the point of view of a Participant in the right)

From this point, neither the AVG nor the transport agent can switch, and there is a real agreement between the two parties, which is no longer provisional, but that before they have an unlimited opportunity to switch. There are two possible alternatives: one will cause the AVG to change, and the other one with the transport agent. Let's first look at when an AVG can choose another transport agent - this case is illustrated by Figure 3.

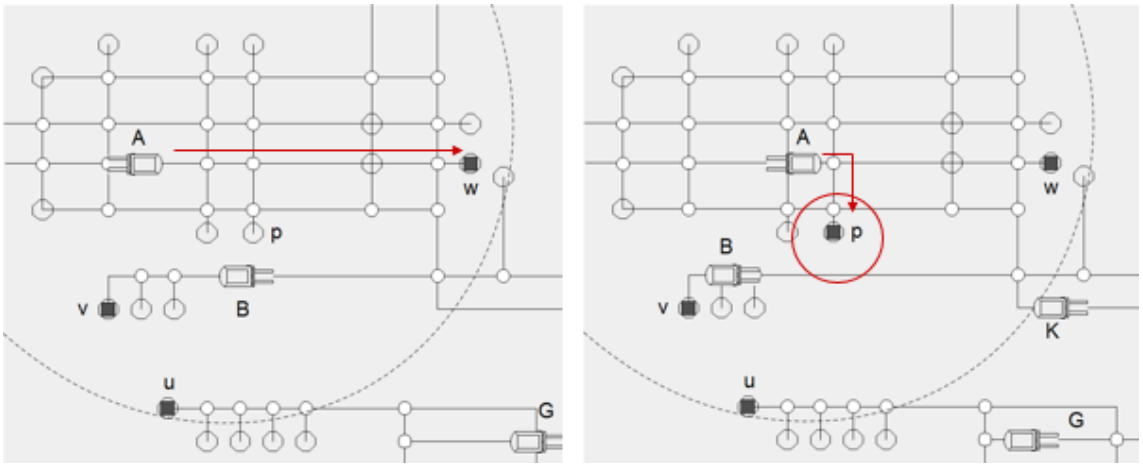

Figure 3: The AVG switch participant because a new better task became available. (Original transport agent of the AGV - in the left, The new trasport agent of the AVG - in the right side

In this case, as seen from AVG on the provisional agreement, W transport agent is the client and is on the way to pick up and deliver it. However, in the meantime, P transport agent appears, who, with a local broadcast message, will make a better offer for the AVG, so A AVG will switch and new client will be the P transport agent.
The other case is for switching when the transport agent gets a better deal in the meantime than it had received from an AVG approved earlier with provisional acceptance. It's important to note that AVG, who has made the previous bid and got the provisional acceptance, is already on the way to the transport agent, so his offer has improved as compared to the offer previously made, because the given that bid estimates are based on the distance between the current AVG and the

transport agent. So when a new AVG appears and sends a bid to a transport agent who already has a provisional AVG, then the new bid should not be compared to the old provisionally accepted bid, but two new and current bids from both agents should be requested.

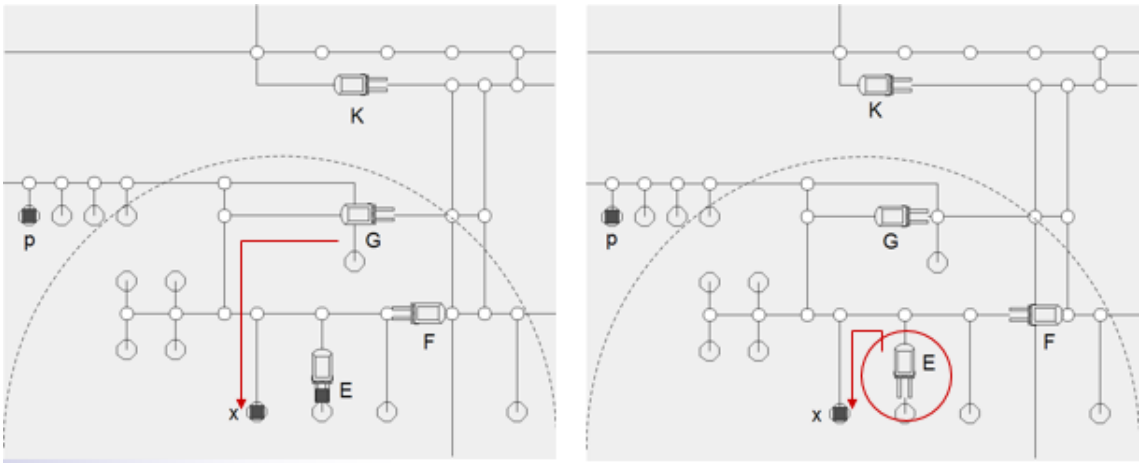

Figure 4: The transport agent switch participant because a new better AVG became available. (Original AVG of the transport agent - in the left, The new AVG of the transport agent - in the right side

This process is visible on Figure 4. X transport agent will have its original provisional acceptance given its AVG G, which will start moving towards it, but during this time, E AVG, who will bid for X, will appear. X considers that G's current position or E's current position is more advantageous - and since E is closer he accepts his offer and will switch AVG.
Due to DynCNet's ability to switch customers from both sides, the protocol is much more flexible and AVGs can achieve better utilization and transport agents will have a reduction in waiting time.

3 Multi-agent system design

During the project, I simulated the operation of a taxi company, which included taxi agents, travel and taxi stations. The system presents the problem of dial-a-ride problem (DARP) and provides a solution.

3.1 Design

In this DARP, the customers want to reach a destination from a particular location. Taxi agents will receive their request and will sign an agreement with CNet or DynCNet communication protocol - and the best bid on AVG's rivalry between taxi agents will win the deal. The aim of the company is to have the minimum waiting time for the customers, to maximize the utilization of taxi vehicles so that as many passengers as possible can travel from A to B over a period of time - which is maximizing revenue. Taxi base stations will be in charge of fuel supply.

3.1.1 ACL messages

Since in my project during message passing there is a direct exchange, I need a common language. During communication, I have chosen Agent Communication Languages (ACL) as a common language, which supports intentional communication. The ACL language is also suitable for this task since it does not define the communication protocol. I structured an expanded version of the language based on [3].

3.1.2 Communication protocols

The protocols I have chosen for the communication and the task allocation were Contract Net and the Dynamic Contract Net Protocol. However, the two protocols are similar in many respects, however, DynCNet can adapt to dynamic changes in the environment, thus achieving better vehicle utilization and even lower waiting time. As I have described in my research work, the great advantage of these protocols is that each agent does not have to have complete information about the entire network - in fact, one agent alone could not solve the task. The essence of the protocols is that it can interact with messengers by messaging exchanges - thus achieving an efficient allocation of resources.

CNet protocol

During the CNet protocol, customers who do not yet have a transport vehicle send a local broadcast CFP message that can be received by taxi agents within their communication range. Customers will be waiting for response from the taxi agents for CFP messages, but there is a time limit for customers to answer the responses - this is required because all the taxi operators who are within this communication range will send a response. A taxi agent can send two types of responses to a customer depending on whether or not the agent has previously agreed with another customer: i) if he has not yet agreed, it will bid to the customer that he will do with a PROPOSE ACL message. It addresses to the customer and also sends its the bid - which is the shortest distance between the taxi agent and the customer ii) if the taxi agent already have an earlier agreement with another customer, it will rejects the request with a REFUSE unicast ACL message.
If the customer receives only refusal messages, it will attempt again until it will receive a PROPOSE message - if it has received bid or bids, it selects the most favorable bid that will be the nearest taxi agent and send an ACCEPT PROPOSAL unicast message for that agent. The other taxi agents who also sent a PROPOSAL message will receive a REJECT PROPOSAL message so they will go on the map looking for another possible customer.
If a taxi agent receives an ACCEPT PROPOSAL message, an AGREE message tell to the customer that the task has been taken, then the agent will move to the pick up location of the customer and bring the it to the delivery location. During this time, the taxi agent will not receive messages from other customers, so if it is receiving a CFP message it will answer a REFUSE or if two or

more negotiations are negotiated at the same time and the other customer(s) has accepted the taxi agents offer as well, then the taxi agent accepts the first customer and send a FAILURE message to the rest.
Once a taxi agent has taken the custom to the delivery location, it will be re-released and will receive CFP messages again and the process will begin.

DynCNet protocol

The difference between DynCNet and CNet is that customers and taxi agents can also switch if they a get better deal - as long as the taxi agent does not reach the customer pick up location and the customer has failed. Communication also begins with a CFP message from a customer who does not yet have a vehicle to ship. Taxi agents check at that time that there is already a customer with whom they agreed on a provision accept. If it does not have an earlier agreement, it will return a PROPOSE message to the customer and submit it with bid. If the taxi agent is already moving to another customer and has a provision agreement, consider that at that moment the old customer or the new one is closer to it - if the old customer is nearer, it rejects the new one with a REFUSE message and moves past the old customer Pick up location. However, if the new customer is closer, the FAILURE sends a message to the old customer - who acknowledges that there is no agent yet to react, so he starts the process with a CFP message - then sends a PROPOSE message to the new customer and sends a bid. If a customer receives bids from taxi agents, as with CNet, it accepts the most favorable offer with a PROVISIONAL ACCEPT and rejects the rest.
If PROVISIONAL ACCEPT is received by an agent, like the CFP message, if it has not previously been accepted by customer, if it has, than it will reviewed whether or not it has received a better offer and decides accordingly.
If the taxi agent arrives at the provision customer's pick up location, it sends a BOUND message that it has arrived and the customer responds with an ACK - after this point none of them can switch partner. Once the taxi agent has reached the customer delivery point, the process begins for it.

3.1.3 Additional constrains

There are various limiting factors involved in the project, which complicate the communication of nodes and the agents situation to solve their problem.
One of the limiting factors is that both the customers and the taxi agents have a limited range of communication so the customers can not reach all the free taxis on the map. As a result, customers who are waiting at the border of the map are likely to have to wait longer. To avoid this, taxi agents who are just free do not wait at one place, but place random targets on the map and go there to find a customer they can take away.

Another limiting factor is the battery of taxi agents. Each taxi has a battery that is constantly decreasing, and if it reaches a level, it has to go to the taxi station and fill up. The taxi drivers will choose the closest taxi station on the basis by the implementation, and if a customer is not already in the car - just the taxi is on the way to pick them up, then it will refuse the customer for refueling. This limiting factor obviously takes a lot of time from taxi agents - it reduces the taxi vehicles utilization and increases the average waiting time for customers.

3.1.4 Simulation and Visualization

The program simulation and visualization was based on the RinSim JAVA simulator and framework.

3.2 Comparison with existing MAS

During my research, I have found similar projects where CNet and DynCNet have been studied [4, 5] - sometimes complemented by comparison with FiTA. Similar projects were also studied in these independent variables during the measurements as mine, such as increasing the number of agents and the simulation time frame. The fact that previous research has examined these issues - from a different point of view or through another example, I think it is because these are the most interesting issues in such a work, how these things will respond to system implementation.

4 Experiments

During my experiments, the main question was the differences between CNet and DynCNet. In the following chapter, I will make explain the setup what I used during the experiments, the results and evaluation of them.

4.1 Setup

During the measurements I used the simulation environment with the same setup for both CNet and DynCNet so that I can compare the two protocols with relevant information. Starting from the fact that customers and taxi agents need to stay in a certain ratio - as there are too few customers in comparison with taxis, the use of taxis is not reduced by the failure of the protocol, but when there is too much then the waiting times of the customers are increased significantly.
Experienced, after adding the battery supply restriction to the experiments, I found 20 customer and 30 taxi agents working correctly, where 0.02 newcomers were likely to emerge. This gave us a fairly well-balanced map, where taxis were able to meet customer needs. There were 8 depots on the map for taxis to charge battery. During my experiments I did not change this number - I considered the length of the jokes as an independent variable that influenced my dependent variables. I made 10 different measurements with both communication protocols, in which the number of ticks was between 1000000 and 10000000, with a step of 1000000 between them.

4.2 Results

During the experiment I compared the utilization of taxi vehicles, the message sent by the protocols, and the transported passengers for the two communication protocols.
In terms of messages, as expected, DynCNet had a lot more messages between devices than CNet, but the difference remained linear. The result of the message rate is shown in Figure 5.

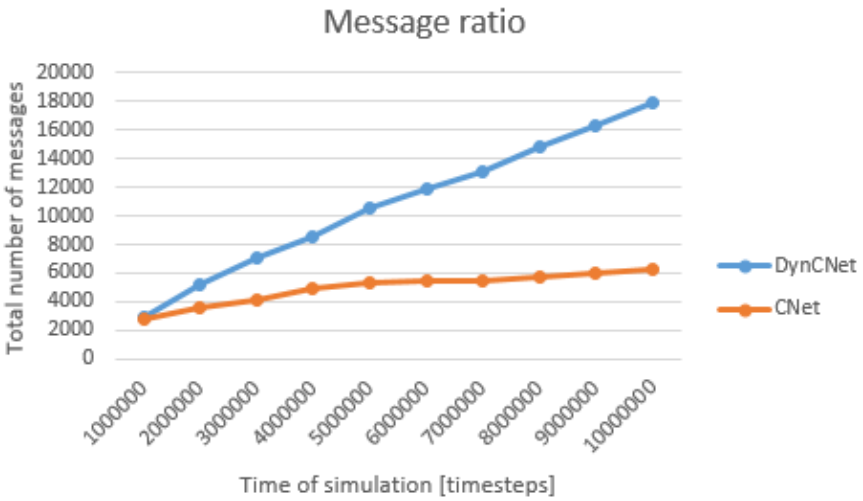

Figure 5: The message ratio between CNet and DynCNet protocol.

The number of passengers delivered during the simulation was also measured. This was more complicated in both cases by the fact that taxi vehicles had to go to the Taxi base to charge the battery.The graph of the result obtained in the simulations is shown in Figure 6.

Figure 6: The number of delivered passengers during the simulations.

Last but not least, I measured the utilization of taxi vehicles, examining how many percent of the cases the passenger is in the vehicle during the simulation and how many percent when the vehicles are free. The pie charts that shows the measurement data are in Figure 7.

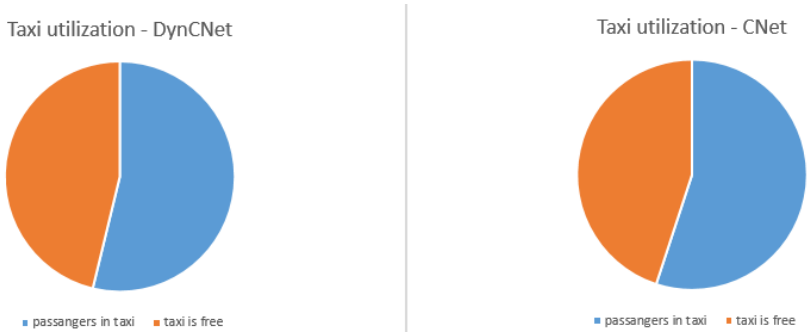

Figure 7: The utilization ratio of the taxi vehicles in average.

4.3 Analysis

In this subsection, previous measurement results are evaluated. During my research, as I mentioned above, I found similar experiments with similar problems with these communication protocols - but some results were different. I will also examine the reasons for these differences, and I will also evaluate the hypotheses that were previously set up, and then examine whether the goals set at the beginning of the project were met.
First of all, the measurement results were the number of messages, which completely give the expected results. For DynCNet, the number of CNet messages has multiplied from the beginning and the gap has been increasing strictly monotonous - but only linearly.
By contrast, the number of passengers delivered during the simulations - and in this context, the utilization of taxi vehicles has came up with an unpredictable result. In other research, DynCNet overperformed the CNet protocol, but in my measurements, the two protocols produced a completely similar result. This is partly due to the fact that since the difference between DynCNet and CNet came when a passenger appears on the map, or a taxi agent gets free again. I think in my experiments these two possibilities did not happen often enough to make a significant difference

between the two protocols. In those cases when I increased this number, my taxi agents could no longer serve the users properly - so that would not have been a solution in this situation. In this context, the utilization of taxi vehicles in the two measurements was also fully the same - since both the number of passengers is transported and also the service duration is the same amount of time, the utilization is also the same.

Evaluation of the set hypotheses for the results:

1. **The DynCNet gives a better utilization for the vehicles than CNet protocol:**
   Since I could not prove this hypothesis, this is a null hypothesis - based on the results, DynCNet has made a slightly better result, the difference is not as significant as I expected.

2. **We can have more finished transport using DynCNet than using CNet:**
   As I described above, these two hypotheses were related and the consequences of each other, so this hypothesis was a null hypothesis as well, because the number of passengers delivered during DynCNet and CNet simulation was practically the same.

3. **DynCnet always generates more messages regardless of time and the number of agents compared to CNet:**
   With regard to the number of messages, DynCNet has indeed made a significant difference, so this hypothesis is an alternative hypothesis as it has been backed up.

4. **Both algorithms increase efficiency by increasing the number of taxi agents:**
   This hypothesis is conditionally fulfilled, but the number of waiting passengers, and the likelihood of their emergence, should be increased as if only the number of taxi vehicles would be increased by itself, the vehicles would be unnecessarily and without work that would not increase efficiency. However, if there is are enough customer to serve, the hypothesis is true, so it is an alternative hypothesis.

5. **Both algorithms increase the utilization of vehicles by increasing the number of customers:**
   Likewise, as the previous hypothesis will be fulfilled conditionally, as the number of customers and vehicles is related to each other. In this case, if we increase the number of customers for a while, the hypothesis will be true and the utilization of the vehicles will increase, but if there are significantly more users than vehicles and the vehicles reach their maximum utilization, then as many customers can be more on the map, the value will not increase further.

Summing up the project, the goals originally set were met:

- ✓ Implement the communication protocols such as: CNet and DynCNet
- ✓ Investigate the efficiency and the performance of these communication protocols
- ✓ To show how the system response time depends on the number of agents and the deadline value with the different communication protocols

5 Conclusion

When evaluating the results, however my results are differ from the expected results, I would still choose DynCNet's protocol if I had to use a protocol in my system. In truth, there is a lot more energy involved in implementing DynCNet and it has more possibility for the developers to make a mistake than in the case of CNet, but assuming that we already have working protocols, I find DynCNet more reliable and more suitable for real-life protocols.
In the real world, the environment is constantly changing, so dynamic communication is required at the level of communication protocols, rather than having a central control system for multi-agent systems that can see the change in the environment, but the nodes themselves have to crawl this situation to help each other.
Since I have not developed a multi-agent system before, it was a very exciting task for me - also to understand the solutions written in the literature, and to implement the protocols. Furthermore, the visualization of the simulations, and the fact that I could see the project during operation was a special motivation.

References

[1] T. Sandholm, "An implementation of the contract net protocol based on marginal cost calculations," in *AAAI*, vol. 93, pp. 256–262, 1993.

[2] D. Weyns, N. Boucké, T. Holvoet, and B. Demarsin, "DynCNET: A protocol for flexible transport assignment in AGV transportation systems," 2007.

[3] A. C. L. Fipa, "Fipa acl message structure specification," *Foundation for Intelligent Physical Agents, http://www. fipa. org/specs/fipa00061/SC00061G. html (30.6. 2004)*, 2002.

[4] D. Weyns, N. Boucké, and T. Holvoet, "DynCNET: A protocol for flexible task assignment in situated multiagent systems," in *Proceedings of the First IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, pp. 281–285, 2007.

[5] D. Weyns, N. Boucké, T. Holvoet, and B. Demarsin, "DynCNET: A Protocol for Dynamic Task Assignment in Multiagent Systems.," *SASO*, vol. 7, pp. 281–284, 2007.