Coursework Report

Peter Stefan
40489803@napier.ac.uk
Edinburgh Napier University - Multi-Agent Systems (SET10111)

Abstract

This paper presents a double auction solution for resource allocation problem in edge computing. A multi-agent system is used to simulate the auction mechanism. In it three different types of resources are offered by server agents and allocated to device agents, bidding on these resources. The auction mechanism is designed to satisfy Economic Efficiency, and social welfare in the system, bidders who bid higher in the auction should be prioritized as well as sellers who are willing to sell at a lower price. Through experiments it is proven that the mechanisms used create a market that functions as expected and performs in a way a real market would in terms of supply and demand effecting the prices of items. The scalability of the system is also proven to be good when measured in the number of messages passed between agents, as it scales linearly with the number of agents.

1 Introduction

Edge computing Edge computing is the practice of moving data over communications networks, for the purposes of distributing computational problems to physically closer machines to the client. This paradigm aims to decrease latency and maximize efficiency, by distributing the problems to edge servers closer to the end device rather than further away in cloud computing which introduces latency, or lag. The communication distance is greatly reduced between server and customer in edge computing making that latency as low as possible and this allows for the implementation of real-time applications, since this also improves processing speeds greatly. Edge computing can also provide more security since the data is processing is decentralized, in the edge nodes. The allocation of said edge nodes, and other resources the edge computing network may provide however, can be difficult. Since these resources are scarce it must be decided which device can get access to which recourse or server where it can offload its tasks. This introduces a competition between devices who want to use the services and resources. This is most easily solved by using auction mechanisms to determine winners based on the preferences and circumstances of the user devices and servers. [1] [2] [3]

Multi-agent system In this paper we simulate the resource allocation issue in a multi-agent environment. This will help demonstrate the competitive nature of resource allocation, each device and server can be represented by agents, who try and maximize their own utility. Device agents winning an auction and receiving resources needed for their computation, and server agents selling their resources for profit. A multi-agent system is perfect for simulating an auction scenario such as this, since the devices, servers and the auctioneer can all be represented by agents, each autonomous, with their own separate goals, the interactions between them modeled by the message passing system. The auction system requires the participants to make decisions, which include the of setting their asking and bidding prices, which represents their utility score for a certain unit of resource they would like to acquire. Thus a complex problem such as this can easily be modeled in a multi-agent systems [1] [4].

2 Design

In the prototype there are three agents implemented. A server agent, and a bidder agent which represent the server holding the resources and the device, bidding for them; there is also an auctioneer handling the auction and receiving the information about the other two: their valuations/prices and preferences for the bidders, offered number of items for the servers. [5] The agents only communicate with direct messages between each other, eliminating any use of broadcasting greatly decreases message passing between the agents, thus improving performance especially if the system is scaled for more agents. [6] An example of reduced message passing would be when and agent wins or loses the auction, the auctioneer directly sends them a message about this, rather than broadcasting to all agents available. The messages are processed in the order they are sent out in the system making the outputs clearer and this also means for the auction mechanism, that time is a factor of determining priority of winners. Older bids are prioritised to newer ones, in the implementation we can assume that $agent^i$ has an older bid compared to $agent^{i+1}$. But this priority only applies if the prices are matching between two bidders, since valuation is the chief priority [7] . This model is represented in a discrete environment, since our intelligent agents can only do a predetermined number of things, they can participate in the auction, give their valuation and their inventory of resources they need or sell, and either win or lose. They are intelligent agents, since they participate in the auction exhibiting social behaviour.

2.1 Auction mechanism and reasoning

The model presented in this paper was created using a double auction protocol to simulate the interactions of the resource allocation issue. This was chosen because it is an auction with multiple sellers (these are the servers) and multiple buyers (these are the devices) each submitting a bidding or asking prices for the various resources, thus it has a many to many structure.[1] It is able to simulate this rather complex scenario. In the system the bidders and sellers both submit their valuations to the auctioneer, who then orders the bidding prices in decreasing and the asking prices in increasing order. The break-even index will then be calculated as k, the largest index where the bidding price is greater than or equal to the asking price:

$$p_k^a \geq p_k^b$$

p_k^a meaning the k^{th} place asking price and p_k^a meaning the k^{th} place bidding price. And from this value the transaction price p^* can be calculated as the average of the break-even bid and ask price:

$$p^* = (p_k^a + p_k^b)/2$$

The auction itself starts via the auctioneer asking both servers and bidders for their information, then these send back their prices and quantities for the different types of items they sell, or would like to buy. In our prototype, each bidder has a units of bandwidth, RAM capacity, CPU capacity it requires to achieve their computational task. In our model we assume that a device agent would not split up their allocation of resources between multiple servers, since this could diminish the benefits on computing speed and lack of latency they achieve by offloading their tasks to edge servers. [8] Thus the device agents will simultaneously bid on all three and the servers will have a predetermined amount they offer at the start of the auction. Both the bids and ask prices are assumed to be the valuations of the agents for one unit of that item. This is the price used in the previously mentioned break-even price calculation to determine the winners. The highest bidders who bid up to the break-even price will be the winners of the auction, along side the servers, who had offered there items for a unit price up to this k index. The winners are matched with the servers, in a way, in which lower server offers and higher bids are prioritized.[7] Then they are further filtered against the matched servers to see if the servers have the required resources to satisfy the buyers. All of the winning bidders will pay the transaction price $p*$ and receive the bought items. Those who lost the auction have to pay 2 times the transaction price for each unit they need, this is to demonstrate these devices having to send their computational problem to a centralized cloud, and paying a penalty for this. The bidders with the highest bids are prioritized thus the system makes it economically efficient (EE) meaning that at the end of the auction the items will be in the possessions of those who value them the most. [7] The mechanism also satisfies individual rationality (IR), since all the winners will be satisfied at the end of the auction, they will not sell/buy, below/above their respective valuations, thus they do not lose utility by participating in the auction. The auctioneer does not engage in the transactions, they do not lose nor acquire any money by the end of the auction, thus it has a strong budget balance (BB). [9] The main drawback of the proposed design is not fulfilling the last economical property truthfulness (TF). In the prototype because of the usage of the break-even price to determine the winners, and the transaction price, a trade-off is made between TF and efficient allocation of buyers.[10] If the device and server agents in this mechanism would like to maximize their utility, their monetary gain, they can do so, by untruthfully reporting, higher (bidders) or lower (sellers) prices than their real valuation. This property of the system is somewhat counter acted, by the fact, that the implementation is a single-round auction, meaning the bidders and sellers both make only one proposal for their valuation, and they don't know the other agent's intentions on this price, thus somewhat negating the negative effects of agents trying to report false valuations. The buyers and sellers both set their valuations randomly in a range, this is can be changed between runs of course. Because the auctions are separate and single-round, meaning every agent only submits one price, this randomness is introduced to demonstrate the deviation in prices between auctions, to make the auctions more fair, in the sense that agents do not know what others have bid, and offered. This is also a measure done to counteract the previously mentioned TF downside of the mechanism used.

Another auction mechanism that was considered is a combinatorial auction mechanism, where the bidders would bid on a combination of items and these would be allocated by the auctioneer in the most efficient way possible based on the agent's preferences. The price would be calculated by the combinations bought, by winners. [1] This could have been a good choice because of the multi item nature of the problem, however there are downsides to using combinatorial auctions. They are not as suitable for multi-user and multi-server auction where as mentioned above we made the assumption of treating tasks as individual atomic tasks, meaning they can not be divided between servers, so they are treated as one commodity in the auction. [11] And even if we treated them otherwise, the combinatorial auction would have a much more challenging implementation, as creating a truly efficient algorithm for matching buyers and sellers can be more complex because of their relationships and personal requirements. This would especially be a problem if the system was scaled with more servers and more users. [1]

3 Evaluation

In the following section we will look at two experiments to demonstrate the efficiency of the system in different scenarios. In the experiments the number of agents in the system will vary, as well as their price and bidding choices. The auction will be run 10 times in each experiment to account for the random valuations and prices of the agents. But first an investigation into the performance as the number of agents is scaled. This is demonstrated using the number of messages passed between agents in terms of the number of agents. As mentioned in the design, message passing minimization was a goal of the system, thus the agents communicate only when necessary and only using direct messages throughout. [6] The experiment will consist of increasing the number of agents and counting messages passed between the agents. If our system only uses direct communication the number messages should scale linearly with the number of agents. (Figure 1.)

The tests show that the number of messages does indeed scale linearly, which means the system can be effectively scaled to more users due to minimizing the messages passed, and these will not have an unnecessarily negative effect on the performance. This also means the proposed system could me more easily implemented for a real world application and the scaling is as good as it can be in terms of message passing.

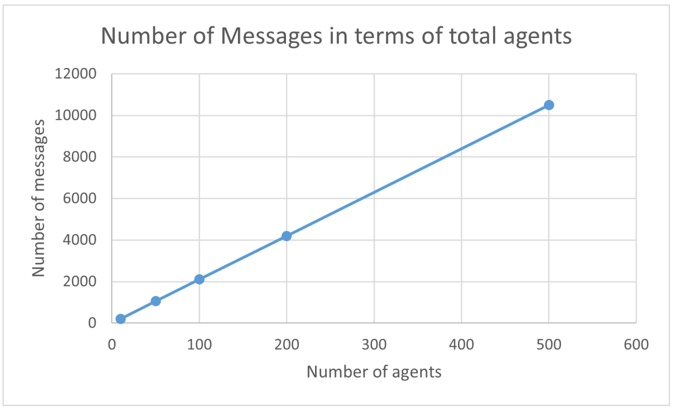

Figure 1: **Message scaling** -

3.1 Experiment

The experiment is done to measure the average success rate and average break-even price in the system as the number of bidders scales up. We start with 50 servers, the number of bidders is scaled from 10, 50, 100, 200, 500, 1000. Each average is then calculated from ten runs to compensate for the randomness of prices and item quantities. This experiment should show the balance between supply and demand shift as the ratio of bidders increases in contrast with the sellers. Our hypothesis is that the average break-even price for the items will increase, since the demand is higher and supply stays the same, only bidders with the highest valuations will win the auction, thus the rate of successful transactions should decrease at the same time, as there is not as enough supply offered. The average break-even price will show if it increases that the items are sold at a higher price at the end of auctions, thus the profit of sellers becomes higher, the profit can be calculated as the difference between the asking price and the transaction price of a seller, this should get closer and closer to the maximum price in the system (the bidding and asking prices are set randomly in a range between [30..50]). Profit of sellers:

$$p_p^a = p * - p^a$$

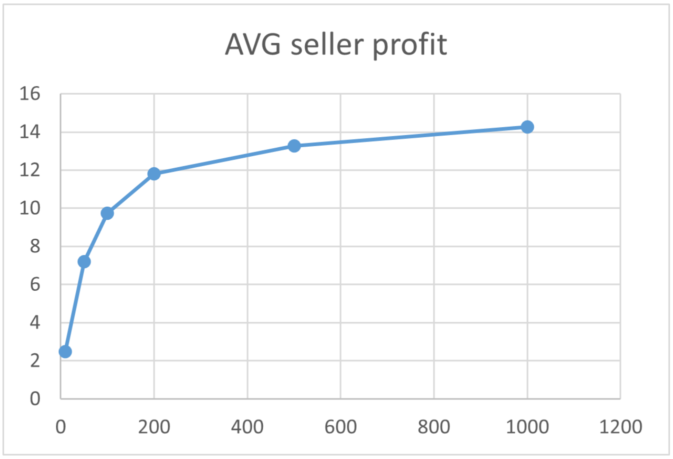

Figure 2: **Profit of sellers** - y axis shows average profit of sellers, where x axis is the number of bidders

The findings of the experiment show that the ratio of successful transaction will scale along a negative exponential curve,

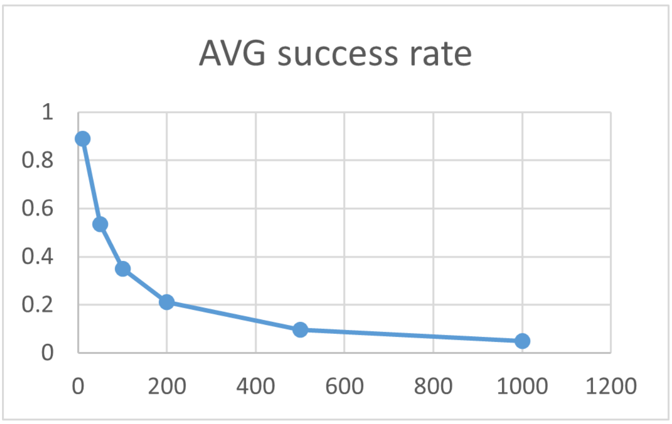

Figure 3: **Ratio of successful transactions** - transaction success compared to number of buyers

meaning it will have a sudden drop as the number of bidders is increased and then it plateaus out. The sellers profit however shows a hyperbola curve as the demand scales up and the supply stays the same, the extra profits accumulated by the sellers grow rapidly at the beginning and slows down as it approaches a maximum possible profit. It approaches a maximum because in our prototype the prices are in a fixed range, and there is a maximum profit that can be achieved. In a well functioning market it is expected that a change in the ratio of supply and demand should cause the price of the commodities to shift consequently as well, and our mechanism showing similar tendencies means it works in effectively simulating such a market.

4 Conclusion

In summary, in this coursework a solution for resource allocation in edge computing using a multi-agent system and auction mechanisms was proposed. The double auction protocol used in the implementation simulates a competitive environment where device and server agents take part in the resource allocation. The agents in the system can bid on and sell multiple resources simultaneously. The winners are matched together using the break-even index of the prices, making the algorithm economically efficient at allocating the resources prioritizing the highest bids and lowest asking prices. However a trade off is made in terms of truthfulness, but the design decision of making it a single-round sealed bid auction is implemented to counteract agents strategically fixing their prices. Future work could focus on making the auction multi-round while still having some kind of mechanism in play to keep truthfulness as high as possible. Exploring combinatorial auctions further could also be interesting as it was considered during the design phase with its main downsides being its complexity. Implementing adaptive bidding strategies and making the agents more intelligent would also be needed in the case of multi-round auctions. This could be done by the device and server agents both having a memory of the previous rounds and depending on if their goals were achieved they could modify their bidding strategies. However, for this system to function efficiently and fairly the design would need to deal with the problem of untruthful bids and asks.

References

[1] H. Qiu, K. Zhu, N. C. Luong, C. Yi, D. Niyato, and D. I. Kim, "Applications of auction and mechanism design in edge computing: A survey," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 1034–1058, 2022.

[2] K. Surya and V. M. A. Rajam, "Novel approaches for resource management across edge servers," *International Journal of Networked and Distributed Computing*, vol. 11, no. 1, pp. 20–30, 2023.

[3] S. Carlini, "The drivers and benefits of edge computing," *Schneider Electric–Data Center Science Center*, vol. 8, 2016.

[4] M. Schumacher, "Multi-agent systems," *Objective Coordination in Multi-Agent System Engineering: Design and Implementation*, pp. 9–32, 2001.

[5] F. Leon, "Actressmas, a. net multi-agent framework inspired by the actor model," *Mathematics*, vol. 10, no. 3, p. 382, 2022.

[6] V. K. Vijay, H. Sheikh, S. Majumdar, and M. Phielipp, "Minimizing communication while maximizing performance in multi-agent reinforcement learning," *arXiv preprint arXiv:2106.08482*, 2021.

[7] D. Friedman, *The double auction market: institutions, theories, and evidence*. Routledge, 2018.

[8] D. H. Roh and H. S. Yang, "Design of an iterative multi-item double-auction mechanism," *Computational Intelligence*, vol. 24, no. 1, pp. 62–75, 2008.

[9] R. Colini-Baldeschi, B. d. Keijzer, S. Leonardi, and S. Turchetta, "Approximately efficient double auctions with strong budget balance," in *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pp. 1424–1443, SIAM, 2016.

[10] W. Sun, J. Liu, Y. Yue, and H. Zhang, "Double auction-based resource allocation for mobile edge computing in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4692–4701, 2018.

[11] X. Lyu, H. Tian, W. Ni, Y. Zhang, P. Zhang, and R. P. Liu, "Energy-efficient admission of delay-sensitive tasks for mobile edge computing," *IEEE Transactions on Communications*, vol. 66, no. 6, pp. 2603–2616, 2018.