Introduction to Multi-Agent Systems
Activity 2

Team 5

Victor Gimenez Abalos - victor.gimenez.abalos@est.fib.upc.edu
Daniel Felipe Ordonez Apraez - daniel.ordonez@est.fib.upc.edu
Albert Rial Farràs - albert.rial@est.fib.upc.edu
Jordi Armengol Estapé - jordi.armengol.estape@est.fib.upc.edu
Joan Llop Palao - joan.llop@est.fib.upc.edu

December 4, 2019

Contents

1 Introduction

In this activity, we will describe the different cooperation mechanisms in multi-agents systems, and how they apply to our particular case. For all the section, apart from the slides of the course, we have relied on [1] as the main source of information.

2 Cooperation Mechanisms

2.1 Auctions

Implementing auction-based collaboration in this work is not as strange as it may sound at first. In auction based protocols, auctioneers offer something that multiple bidders want to attain in exchange for something. In the case of our MAS, the closest resembling structure would be the relationship Manager-Classifier agent. However, we require two other things other than the structure, that is, what is exchanged. In the case of this classification task, we believe what the bidders should want is to get credit from a prediction (i.e. the manager reports their prediction to the user). On the other hand, what the classifier in question bids would be 'credibility' or 'trustworthiness'. This mechanism could work since most ML methods offer probabilistic answers, which means that the agent will 'more or less' know how certain he is about a prediction. So, the way this mechanism would be implemented, would be: for each prediction, the auction starts, the classifiers bid credibility, in exchange for having their answer as the reported one. Our manager agent was originally designed as a hybrid model. This means that, we could use the deliberative/proactive mechanisms to have the manager offer 'fake' auctions to establish the credibility of the agents. Credibility, in turn could be used for several things:

- Killing/re-training low-credibility agents, since their predictions are bad.
- Using credibility as a weight in a hybrid mechanism with voting.
- Detecting classifiers which are not bidding often (indecisive), possibly for re-training them.

For any of the possible auctions, we could consider bidding for instances of the dataset individually, for the whole set, or via a combinatorial auction. However, this is highly dependent on the actual target of the platform: if real-life predictions are going to be individual (for example, the fertility of people will most likely be predicted one person at a time), then the first option would be both simple and not too inefficient. We debate the pros and cons of each: While the first two would be easier to implement, the first one would be too computationally expensive, and the second one would penalise specialised classifiers (since they would have high confidence for small subsets of data and low for the rest). The combinatorial one would be the most complex yet best option in a generic case. We would need to worry about how to implement bundling of items on the classifiers, which has several good approximations (thresholding), about the Manager assignation of winners, and also about those instances of data that no classifier bids for. As for the different kind of auction mechanisms, we list a couple of possibilities below:

2.1.1 English Auction

Each classifier receives the prediction data, and then starts bidding credibility. Each classifier would set as upper bound of their credibility (i.e. their confidence) something that is a function of their predicted probabilities (like the most likely minus the second most likely, or the probability of the most likely element). The English auction, by design, is quite inefficient, but it does not heavily punish overconfident agents.

2.1.2 Dutch Auction

It would be the same idea as above. However, this time, only one agent gets to bid: the one that has the highest confidence. This could result in a more efficient auction scheme, since we expect the confidence of the winning bid to be high, yet we would have a high number of messages from the price-lowering of the manager. In exchange, an agent that is overconfident and makes an error would be severely penalised, even if it is usually correct.

2.1.3 FPSB

This auction has the same inconvenient as the dutch auction. However, given that the communication scheme is faster (many less messages between the manager and the classifiers in the mean case). As a matter of fact, the number of messages between agents would be constant, since this is desirable. However, we still heavily penalise the confident classifiers, which, again, is not desirable. Another problem is that

2.1.4 Vickrey

We believe this would be the best auction scheme. As with FPSB, the number of messages would be constant, therefore the communication is more efficient than in the other two counterparts. As for the overconfidence discussion, this format would reduce the impact, since the agents will only be penalised when another agent is also being overconfident.

2.1.5 Advantages and disadvantages

We do not consider any more complex auction, since the ones above serve our purpose.

This mechanism's disadvantages are quite obvious:

1. The verbosity and amount of messages sent make it inefficient with respect to the others, even in its most efficient format.
2. We require of slightly more complex mechanisms (such as the credibility scores) than with other mechanisms.
3. It would only be applied to the collaboration between two types of agents, the rest would require of another mechanism in which this one would be imbricated.

On the other hand, we have several advantages:

1. The credibility mechanisms allow for the BDI paradigm to have autonomy, particularly for the manager agent.
2. Idle times of the system (user is not requesting things) could be actively used to re-compute credibility scores and raise overall accuracy.
3. We would be able to report the credibility and the confidence of any prediction, which would definitely be useful for a decision-support system such as this one.

2.2 Voting

Voting is a cooperation mechanism in which the outcome of the negotiation is based on the votes of the involved agents. In the case of our project, it is quite intuitive to imagine how such mechanism would work with regard to classifiers. Since we have multiple classifier agents, but the user is probably keen on getting a single result, we could use a certain voting mechanism for aggregating the different predictions of the classifiers. In the case of the other agents, we do not appreciate any use case in which it would be useful to implement a voting mechanism involving the other kinds of agents (the user agent and the manager). Voting mechanisms can be applied we have multiple agents, typically of the same kind, but it does not make much sense to use them when we have a hierarchical structure (eg. the manager supervising the classifier agents), or 1-to-1 or 1-to-many relations.

The most generalized view of voting aims to rank a set of alternatives based on the preferences of each individual agent. We want to define a social choice rule that should be calculable, complete (defined for all pair of alternatives), linear (the social preference ordering ¡* should be anti-symmetric and transitive over O, the set of options), anonymity (the outcome does not depend on which specific agents have certain opinions), unanimity (if all agents believe that A is better than B, then A should be better than B in the resulting order), neutrality (the outcome should not depend on the order or naming of the alternatives) and independence of irrelevant alternatives (adding or removing an irrelevant alternative should have no effect on the outcome), although no voting mechanism fulfills all the properties, as stated in Arrow's impossibility theorem.

2.2.1 Voting systems

In the case of the basic voting mechanisms, we can consider the following options:

- **Plurality:** Each agent can give one vote to one of the alternatives, and the preference with the most votes wins. One of the main problems of this option is the useful vote, when voters vote their second-best option because they believe that their first option has no possibilities of winning. However, we believe that this problem does not hold in our case, since our classifiers will not be that intelligent to think in these terms, and actually they have no access to this information (they do not know which options have won in the past, and they do not have access to any poll). On the other hand, problems such as the huge effect of irrelevant alternatives and the fact that only giving one vote gives scarce information about the preferences of each voter will happen. Nevertheless, it is simple to implement and efficient. If all the classifiers are based on the same (or similar) algorithm and the manager does not know its strengths and weaknesses, the equality principle (1 agent = 1 vote) makes sense.
- **Anti-plurality:** Like plurality voting, but the votes are negative, and the option with the less votes wins. The advantages and disadvantages of this mechanism are similar to the ones of plurality voting, but from the reverse angle. In our case, even though the typical use case of classifiers is giving the class with the highest confidence, we could retrieve the class with the less confidence. However, this may not be implemented in Weka for some algorithms. In addition, we believe that very frequently we would observe the case in which all classifiers posited that a certain instance did not belong to a certain class, obtaining a draw. The nuances could be between two classes that no algorithm had negatively voted to.
- **Best-worst:** In best-worst voting systems, each agent gives a positive vote to one alternative and a negative vote to the other one. It is a generalization of the two previous mechanisms, and the importance of the positive and negative votes can weighted with some hyperparameters. In our case, it could be useful for getting richer information from the classifiers, but again we are not sure that getting the class with the less probabilities is implemented for all classification algorithms in Weka. Also, it would be useful in the case of multi-class classification, but it would be a waste of resources in binary classification.
- **Approval voting:** In this case, each agent selects a subset of the alternatives. Each classifier would take the k most probable alternatives. Our datasets are simple and do not have a large number of classes, so probably this would be way too complex for our needs (eg. in ternary classification it degenerates in a best-worst system).

In the case of protocols based on total orders, there are the following mechanisms:

- **Binary protocol:** In this protocol, all the alternatives are ordered and then evaluated in pairs. Notice that this mechanism is not neutral, because the order of the pairings affects the outcome, so in our case the manager would influence the decision (although it could take these decisions randomly). In addition, this protocol does not fulfill the unanimity property. Finally, it has a temporal cost, in the sense that we have to implement the sequential pairwise eliminative votes. For these reasons, we discard this option from now on.
- **Borda protocol:** In this system, each voter assigns —O— points to the preferred option (being O the set of options), —O—-1 to the second best option, and so on. The option with the most points wins. The Borda protocol is the most computationally expensive, and it was some paradoxes that imply the no fulfillment of some of the properties. Therefore, we consider it not worth to implement.
- **Condorcet:** In the Condorcet protocol, each voter ranks the candidates in order of preference and then each candidate is compared to each other. The winner must win all the comparisons, which could entail circular ambiguities.

Finally, the voting systems can be made more complex by introducing the use of linguistic information or the management of uncertainty. In the first case, it is obvious that it does not apply to classifier agents. In the second one, studying the use of uncertainty in voting mechanisms for coordinating classifiers could be interesting, but in our case our classifier agents will be relatively simple, so it would probably imply way more complexity than needed.

If we decide to use voting, we advocate for the most simple alternatives. The protocols based on total orders seem to be more difficult to implement, less computationally efficient and probably way too complex for the nuances that our relatively simple classifier agents can express. In the case of the simple agents, we advocate for the simplest one, plurality voting, for its simplicity, efficiency and the fact that our classification problems do not have many classes and our classifier agents will be relatively simple.

2.2.2 Advantages and disadvantages

We believe that voting mechanisms are suitable for our project in the case of the classifiers. As we said, voting mechanisms can be applied with multiple agents, typically of the same category, but not so much with hierarchical structures or 1-to-1 or 1-to-many relations, as in the case of Voting mechanisms can be applied we have multiple agents, typically of the same kind, but it does not make much sense to use them when we have a hierarchical structure (eg. the manager supervising the classifier agents), or 1-to-1 or 1-to-many relations. In the machine learning literature, there is a topic known as ensemble learning, which studies approaches in order to combine different models into a single, aggregated output. Ensemble learning has been shown to improve the performance with respect to individual models. A particular technique for aggregating machine learning models precisely consists in voting. In addition, since in principle the different classifiers will be in the same hierarchical level and there will be a number of them, voting seems like a very natural fit.

Regarding the disadvantages, we can think of the following problems: Since it is a social choice rule, in principle with anonymity and no dictatorship, the voting system will not account for the fact that some classifiers might specialize and become better in some cases. if an agent is way more confident than the other ones, there is no way to account for this difference in confidence. The social choice rule will not be perfect and we will lose information and nuances when aggregating the votes.

2.3 Coalition Formation

Coalition's formations are collections of individuals/agents working together for the purpose of achieving a specific task, that cannot be normally achieved by single individuals or can be better performed by several individuals united.

The main characteristics of Coalition's formations are:

- Agents within coalitions bring complementary abilities/resources. I.e. there is difference and variance between agents.
- Coalitions are in the general case goal-directed and short-lived.
- There is usually no coordination among members of different coalitions.
- The organizational structure within each coalition is flat.

Now, it is relevant to consider the coalition formation creation process. One of the many challenges of Coalition's formations is that once the set of tasks and agents increases so does the solution space which describes which formations are more optimal for a given task and which combination of formations are best for the global architecture performance. In order to find the optimal coalition formations the following activities are performed:

1. Each coalition (single agent or group of agents) is assigned a value, proportional to the fitness of that coalition for a specific task.
2. Depending on these values, the formations of multiple coalitions are generated in order to maximize their combined value. These are called coalition structures i.e. groups of coalitions.
3. Define how to distribute thee payoff within the structure for each coalition.

The value of each coalition as a measure of its fitness for a specific task is calculated by defining a set of requirements needed to complete the task, and a set of capabilities for each agent, in order to match capabilities and requirements in the agent, coalition, and structure level.

2.3.1 Applicability to the proposed system architecture

Considering this cooperation mechanism for the practical exercise we can relate some of the previously mentioned concepts with the already proposed architecture. In our application, there exist only two tasks/actions: training and prediction, from which only the latter requires a cooperation mechanism in order to coordinate and ensemble the multiple classifier agents outcome on a certain dataset after their adequate training.

Consequently, only the classifier agents have the capability to perform this task. The value that each agent has to perform the prediction task can be measured as a value directly proportional to a metric of performance obtained during the agent's last training (e.g. n-fold cross-validation accuracy, validation accuracy, f1-score, etc). It is important to note that such a capability which will expire once a new training action is required.

Furthermore, with this cooperation mechanism we need to consider the role of the Manager and User agent. For the later one, considering the agent properties and functionalities, there is no direct need for this individual to participate in a cooperation mechanism. Nevertheless, in the case of the Manager agent, although it also does not have a need to participate in the fulfilment of a cooperation task, this agent can be modeled as the "External Agency" which handles the arrangement of the coalition formations structure with the best value/capability to perform the prediction task.

When selecting the best formation structure, there might be a number of classifier agents that could be ignored. This behavior might be desired in the cases where a classifier agent was trained with non-optimal hyper-parameters or dataset partitions resulting in the introduction of error or noise in the classification ensemble. Although this is a likely scenario considering that the human-user has control over each classifier training hyper-parameters and datasets, the architecture proposed for the system assumes all agents are benevolent and thus all agents will cooperate to achieve better performance. Having this in mind, a way to avoid the defective performance of one of the classifier agents, each of them can perform a verification of the parameters assigned to them by the human-user, in search of known hyper-parameters associated with a misguided training process.

Finally, we conclude that this cooperation mechanism is applicable for the proposed architecture and application. Nevertheless, it is not the most appropriate mechanism since the application has a low complexity and scale. Below we summarize the advantages and disadvantages of this coordination mechanism when consider its application this work and in the general case.

2.3.2 Advantages and Disadvantages

We identified the following advantages:

- It offers an automatic way to cluster and select the classifier agents given a metric of value, which can be proportional to the agent classification performance.
- Offers scalability to high-number of classifiers (without considering computational expenses).
- Coalitions formations can specialize in a specific algorithm or application; e.g. regression and classification.

And the following disadvantages:

- This cooperation mechanism is better suited for applications where there is a high number of actions/tasks to perform or a high number of heterogeneous agents.
- Classifier agents will have only one capability; classify. There is only one task requiring cooperation; predict.
- Some classifier agents might not be selected into the final coalition structure. Assuming all agents are benevolent, this is not desired.
- The search space of possible coalition's formations grows exponentially with respect to the number of agents.
- The search space of possible coalition's formation structures grows exponentially with respect to the number of agents.

2.4 Contract Net protocol

2.4.1 Mechanism description

The Contract Net protocol is a high-level protocol that facilitates distributed control over a cooperative task with efficient inter-node communication. It is assumed a loosely coupled system (this means a system in which the agents are connected through a message transfer system), and asynchronous nodes. The main issue to be resolved is how tasks are to be distributed among the agents.

The protocol works as follows:

1. When a task has to be done, the manager of such task sends a task announcement message to all the agents available to execute it. Typically, a task announcement message includes:
   - Eligibility specification (a list of criteria that an agent must meet in order to submit a bid).
   - Task abstraction (a brief description of the task to be executed).
   - The bid specification (a description on the form of the bid).
   - The expiration time (a deadline for receiving bids).
2. Once the agents have received the task announcement, start the bidding process. In this part of the protocol, each agent sends a brief specification of the capabilities of the agent that are relevant to the announced task.
3. When a bid is received, the manager ranks the bid relative to others under consideration. If, as a result, any of the bids are determined to be satisfactory, then the contract is awarded immediately to the associated bidder.
4. Finally, the awarded agent executes the contract. Several scenarios can happen:
   - The contractor can report to the manager that the contract has been completed with a final report.
   - The contractor can report a partially executed task with an interim report.
   - The manager can also terminate contracts with a termination message.

A priori, our system does not match the problem solved by the contract net protocol. Our system executes a batch of instances (classify each instance) in all of the classifiers, then the manager decides which of the results have to be considered as the correct ones. In this approach, we are duplicating the work in all the agents without considering if there is one agent clearly better for the task than another. Also, we are preventing the system to solve more than one batch at a time (for example in the case where several batches are clearly better classified by different agents).

In order to solve these problems, we present an implementation of the Contract Net protocol for our system. Assuming that there is always a way to know the confidence of a classifier with its output and that all classifiers have been already trained, the protocol starts when the user sends a P-order to the manager with a batch of instances:

1. The manager receives a batch of instances to be classified. A small subset of these instances is randomly extracted from the batch and sent to all the classifiers that are available at that moment (the ones that are not executing another task). The task announcement will be as follows:
   - Eligibility specification: The minimum confidence required for an agent to be considered as a possible contractor. This threshold should reduce the number of messages exchanged between the manager and the classifiers.
   - Task abstraction: The subset of the batch instances to be classified in order to know which agent will perform better with the full batch. The chosen subset must represent the full batch.
   - The bid specification: This part will be implicit since the form of the bid will be the standard way of communicating the results of the classification between the classifiers and the manager.
   - The expiration time: We could use this part of the message to specify a limit time to execute the classification of the subset of instances, in case a classifier does not finish in time it is automatically discarded of the bidding process. This would allow us to use our system in cases with time constraints.

2. When a classifier receives a task announcement, the agent classifies the subset of instances attached to the message. If the confidence of the classification result is higher than the threshold and the time to classify the instances has not been too much, we send a bid with the results obtained and the confidence.

3. The bidding process is performed between the classifiers that have sent a bid (the ones that have classified the instances in time and with a confidence higher than the threshold). At this point, we could use one of the cooperation mechanisms previously explained to decide which of the classifiers receives the task (for instance, we could apply a first-price sealed-bid auction). In case that the manager receives zero bids, there are two options:

   - All classifiers have received a task announcement, but none of them has accomplished to send a bid (timeout or not enough confidence). In this case, the best option is to use one of the previous mechanisms by sending the full batch of instances to all the classifiers.
   - Another option is that one or more classifiers have not received the task announcement because they were already working to classify a previous batch of instances. In that case, an option would be to wait until they have finished and send another task announcement to them, and repeat the previous process.

4. Once we have determined which agent will receive the full task, we send the rest of instances to the winner (only the instances that have not been already classified in the task announcement). At this point, several scenarios can happen:

   - The classifier finish the contract and send the final results.
   - There are errors in some of the instances to be classified, then the classifier sends a cancel message to the manager.
   - A T-order is sent, which obligates the manager to send a termination message to the classifier in order to stop the process and start a new training phase.

2.4.2 Advantages and disadvantages

We identify the following disadvantages:

- We are assuming that a batch of instances can be represented by a subset of them. This could not be true, or the subset could be biased, then the result of the bidding process would not be trustworthy, and the classifier assigned with the task could not be the best one for it.

- Another problem is the loss of time when none of the classifiers pass the confidence threshold. In that case, starting the protocol is useless, since, in the end, we are performing another cooperation mechanism.

- In the case where one classifier is the best in several consecutive batches, we could have a sub-optimal result, since the classifier would be first assigned to the first batch, then the second batch would arrive while the classifier is performing the classification process, therefore it could not participate in the bid for the second batch, which would be assigned to another classifier (which could be a sub-optimal classifier). This problem is inherent to the concurrent problem and it is solved in the best way possible (when the best classifier is occupied the second-best classifier is the one who takes the task, and so on).

As far as the advantages are concerned, we consider the following items:

- The Contract Net protocol allows us to implement a solution for the case where the batches of instances to be classified can be represented with a subset of them.

- Another advantage is the energy saved by execution only a small subset in all the classifiers (instead of the full batch) and then executing the full batch in the best classifier.

- The concurrency of executing different batches at the same time in different agents makes this system much more time-efficient than the previous options.

2.5 Implicit Cooperation

The implicit cooperation consists of having cooperation between agents without an explicit exchange of communication messages. This means that the agents do not talk to each other directly. The way they have to cooperate is by modifying the environment. The modifications an agent makes, influence the behaviour of other agents. For that reason, these modifications have to be carefully designed, in order to make the agents contribute towards a useful global behaviour of the community.

Although we could think that implicit coordination is the same as emergent coordination, this is not true. The basic difference between them is that in emergent coordination agents are self-interested, i.e. they do not care about the other agents in the system. Instead, in implicit coordination, the one that designs the system takes care and intends to provoke the emergence of the socially intelligent problem-solving activities.

The reasoning mechanisms that we could apply for our implicit coordination could be thinking about individual agents (methods that allow building a model for the other agents) or thinking about the whole agent's society (methods that try to impose some kind of organization). For our project, we think it could be interesting the second mechanism. In fact, this mechanism has several common approaches such as Social Laws, Electronic Institutions and Organisational structures.

- Social Laws: are global rules which agent follows and that are designed to lead the entire system to a coherent behaviour.
- Electronic Institutions: are kind of a social structure that has norms, procedures and protocols to be followed by the agents and conventions (acceptable and unacceptable actions).
- Complexity: there are agents that are too simple to generate and understand long plans.
- Organisational structures: are structures that define a pattern of information and control relationships between agents, which shapes the types of interactions among them.

Of these three approaches, we think that the one that suits better for our case is the third one, the organisational structures, so we will explain in detail what it is and how we could apply it to our system.

2.5.1 Organisational Structures

As we have said, these structures define a pattern of information and control relationships between individuals, which shapes the types of interactions among them. Specify which actions an agent will undertake and impose restrictions or norms.

These kind of structures involve the creation of roles, communication processes and formal reporting relationships in an organization. The roles identify activities and services necessary to achieve social objectives, not individuals.Each role has rights and obligations that ensure a good global behaviour of the system.

We have three main types of organizations: markets, networks and hierarchies.

- Markets: in markets the main purpose is to exchange some goods. Some agents provide services and others require them. There are also intermediate agents between them.
- Networks: in these structures there are contracts established in a dynamic way between agents and there is mutual interest between them.
- Hierarchies: agents specialise in concrete tasks and collaborate fully with others. Coordination is achieved by using command and control lines.

Our practical work could be seen as a kind of hierarchical structure. Specifically, this structure would be a functional structure, i.e. actors with same role would work together. The roles defined would be: User, Manager and Classifier. Here we present a schema of the mentioned structure (figure 1).

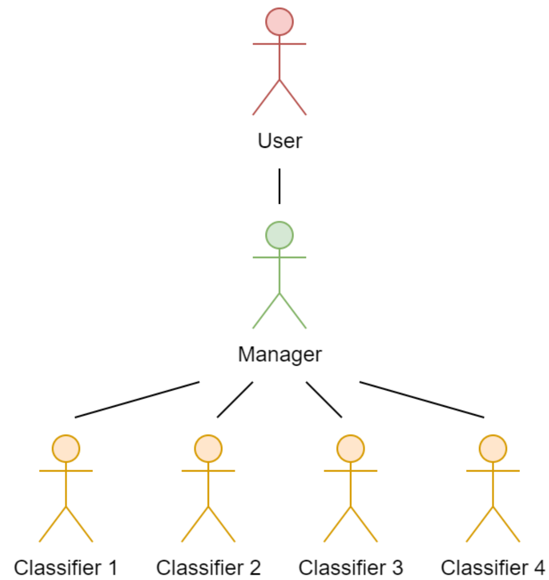

Figure 1: Hierarchical structure

On the top of our hierarchy we have the User agent, who controls what the Manager and the whole system do. This one, the Manager, coordinates all the Classifiers of the system. These agents are specialised actors and can easily share resources between them.

2.5.2 Advantages and Disadvantages

Doing organisational structure (implicit coordination), and specially the last approach presented, a functional hierarchy, we have identified the following advantages:

- Using the structure presented, the classifier agents can easily share resources.
- The classifier agents can work together under the supervision of the manager, who coordinates all their activities.
- Classifiers, as they are specialised actors, can work in tasks reusable.
- As we don't have communication between agents, this mechanism is faster than others.
- Also, as we don't share messages or plans with others, external agents cannot discover what our intentions and plans are.
- Implicit coordination does not need a communication channel, so we could avoid it in our system.

The disadvantages that we have identified are:

- The manager agent has to supervise all the classifiers. However, as in our case the system is expected to be small, we consider this disadvantage is not a real problem.
- As designers, we think that the design of this system would be more complex because we would have to design the organisation (society) and define the different roles and its rights and obligations, taking into account the entire system has to a coherent behaviour.

For the advantages explained before and because we think that there are more advantages than disadvantages, we consider this coordination a good option for our system.

3 Chosen Cooperation Mechanisms

Considering the analysis performed in Section 2 on each cooperation mechanism both for its applicability for the proposed architecture/application, and the general application case, we provide a summary of the suitability of each cooperation mechanism for this work in table 3.

	Appropriate	Applicable but not appropriate	Not applicable
Auctions		✓	
Voting	✓		
Coalition formation		✓	
Contract Net		✓	
Organisational structures	✓		

We conclude that even though all the studied cooperation mechanisms could be adapted to our case, as detailed in the previous sections, not all of them are appropriate. Auctions, Coalition's Formation and the Contract Net protocol can be adapted to the proposed architecture, but these cooperation mechanism imply a considerable effort and complexity in their use, which make their application over-complex. In contrast, both voting and organisational structures are suitable and adequate, apart from being applicable.

3.1 Voting

Voting is a simple way of aggregating the answers of the different classifiers, which are, a priory, of the same category. Apart from being a natural way of aggregating the preferences/predictions of *equal, anonymous citizens*, this cooperation mechanism has been subject of extend studies in the literature analysing their applicability for machine learning applications, an approach known as *ensemble learning*. We discard the most complex voting mechanisms, which are too complex for the classification problem and algorithms present in the current application, and decide to use either, the most simple alternative: plurality voting, or some simple alternatives that still allow the agents to give more than one vote: approval voting or best-worst voting for multi-class classification.

In the first activity deliverable, we considered to implement a mechanism for giving more relevance to the classifiers that had better training performance, but this approach would not exactly be a pure voting mechanism, in the sense that the relevance assigned to each voter/agent, violates the anonymity principle.

3.2 Organisational Structures

In the case of organisational structures, we could use a hierarchy structure where the user controls the operation of the manager and the whole system. The manager coordinates the classifiers, which are specialised in their prediction tasks and can share resources between them.

3.3 Conclusion

The two selected coordination mechanism are not incompatible nor exclusive, on the contrary they can actually complement each other. When using voting, we would have explicit messages between agents, but we could also maintain the hierarchy structure of the organisation in order to have the classifiers fully collaborative between them and the manager coordinating and supervising their activities.

4 E-Portfolio

Our E-Portfolio tool is Github. We use it not only as a repository, but for assign tasks, manage their status and track our meetings.

4.1 Task distribution

The activity has been divided into tasks and these have been assigned to each member of the team, apart from general tasks that have been assigned to the whole group. For holding the tasks we have used the repository in Github (https://github.com/jordiae/IMAS-MAI/projects/8). In the next figure 2 we can see the task distribution done for this activity:

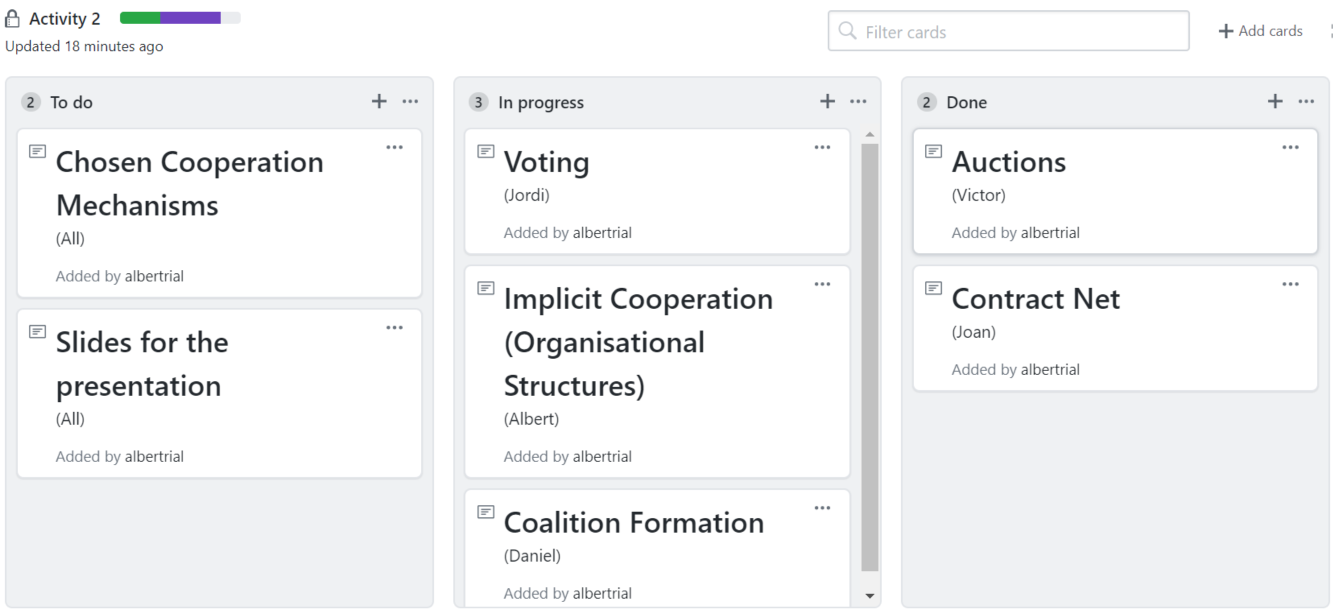

Figure 2: Task distribution using Github

4.2 Project meetings

In order to track our meetings and write the topics discussed and the decisions made, we use the Github Wiki (https://github.com/jordiae/IMAS-MAI/wiki).

For this activity we have done only two meetings. In the first one we divided the work into subtasks and in the last one we discussed about which coordination mechanisms are appropriate for our project.

Below we describe the information about the two meetings done:

4.2.1 Meeting 1

- Date: 28/11/19
- Duration: 2 hours
- Assistants: All team members
- Summary: We reviewed the statement of the activity and we split the tasks.
- Topics:
  - Tasks to be done

- Discussion about the dynamical creation of Agents
- Discussion about our first comments and thoughts about the different coordination mechanisms

- Decisions:
  - Splitting the work into subtasks, working remotely and checking on each other's work.
  - Making a Google Docs in order to keep all the team updated with the last version.

4.2.2 Meeting 2

- Date: 02/12/19
- Duration: 1.5 hours
- Assistants: All team members
- Summary: We reviewed the status of each task and decided which coordination mechanisms are appropriate for our project.
- Topics:
  - Revision of the analysis, advantages and disadvantages of each coordination mechanisms.
  - Discussion about the appropriate coordination mechanisms.
- Decisions:
  - We decided that the best ones we could implement are voting and organisational structures (hierarchical).
  - We decided how we would do the presentation of the activity.

References

[1] Michael Wooldridge. *An Introduction to MultiAgent Systems.* Wiley Publishing, 2nd edition, 2009.