SOW-BKI329 Representation and Interaction (2017-2018)
Practical Assignment 2

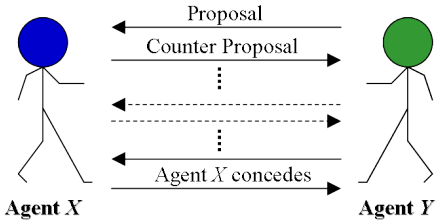
J. Kwisthout and P.C. Groot

Assignment 2: Multi-agent negotiations

In the lecture we saw how agents can negotiate in order to reach an agreement on matters of common interest. In particular, we saw how the Zeuthen Strategy can be used (by both agents) in order to arrive at an agreement. In this assignment you will implement this strategy in the agent-programming language Jason and you will experiment with different negotiation sets. In particular you will explore the advantages and disadvantages of this strategy, as discussed in the lecture, and will investigate the effects of *cheating* in this strategy.

1 The Monotonic Concession Protocol

In order to implement the Zeuthen strategy, we need to be able to represent the Monotonic Concession Protocol (MCP). We need to be able to represent and reason with deals, negotiation sets etc. The first task helps to get used to working with the necessary structures in a agent-programming environment. First we will give a brief recap of the MCP. More details can be found in Section 15.3 of the book and in the corresponding lecture slides.



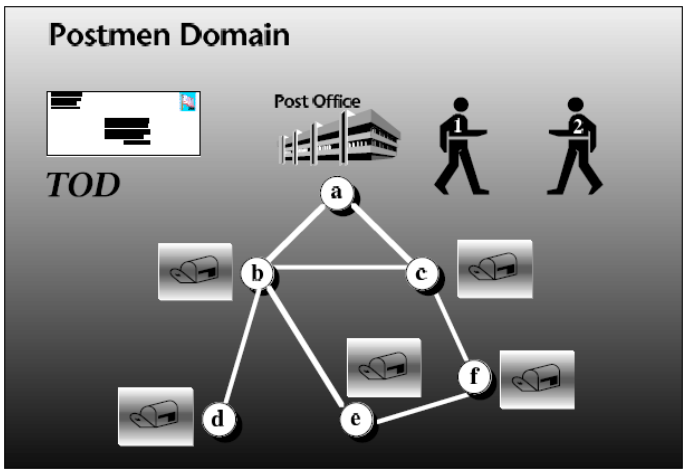
- Negotiation proceeds in rounds. In each round, both agent X and agent Y propose a deal from the negotiation set.
- At round $t + 1$, no agent is allowed to make a proposal that is (strictly) *less* preferred by the other agent than the deal proposed in round t. However, either agent may choose to stay put and repeat the proposal at round t.
- If neither agent concedes (makes a proposal that is strictly *more* preferred to the other agent than the proposal at time t) then negotiations end with the conflict deal.
- If at least one of the agents believes that the deal proposed by the other agent is at least as good or better than its own current proposal, agreement is reached.

The protocol has a few 'degrees of freedom' that are filled in by specific strategies implementing it, particularly the question what each agent should initially propose, whom of the agents should concede, and what should be conceded. The *Zeuthen strategy* implements these questions as follows:

1. Each agent should first propose its most preferred deal;
2. The agent *least willing to risk conflict* should concede;
3. This agent should concede *just enough* to change the balance of risk.

2 Implementation of Tasks, costs, and deals

In this assignment we assume that both agents are two post officers (A and B) that have been given a stack of papers to be delivered. Both agents start at the post office (a) and the cost of their allocated stack T is the length of the minimal path starting at (a), visiting all addresses in T (along the graph), and returns to (a). An example of a graph with five adresses (b-f) is given in the figure. Please use this example as (one of the) running examples in your implementation and report on its results; that facilitates the evaluation of the correctness of your implementation.



Path lengths:

(a,b) =	2
(a,c) =	3
(b,c) =	4
(b,d) =	6
(b,e) =	5
(e,f) =	4
(c,f) =	3

So, if agent A has $T_A = \{b, c, f\}$ and agent B has $T_B = \{d, e\}$ then the respective costs would be $(a, b) + (b, c) + (c, f) + (c, f) + (a, c) = 2 + 4 + 3 + 3 + 3 = 15$ and $(a, b) + (b, d) + (b, d) + (b, e) + (b, e) + (a, b) = 2 + 6 + 6 + 5 + 5 + 2 = 26$.

A *deal* $\delta = \langle D_A, D_B \rangle$ is a redistribution of tasks T_A and T_B such that $D_A \cup D_B = T$ and $D_A \cap D_B = \emptyset$. If $D_A = T_A$ and $D_B = T_B$ then δ is the conflict deal Θ. The *utility* of a deal δ for agent i is defined as $\text{utility}_i(\delta) = \text{cost}(T_i) - \text{cost}(D_i)$. It follows that the utility of the conflict deal is zero.

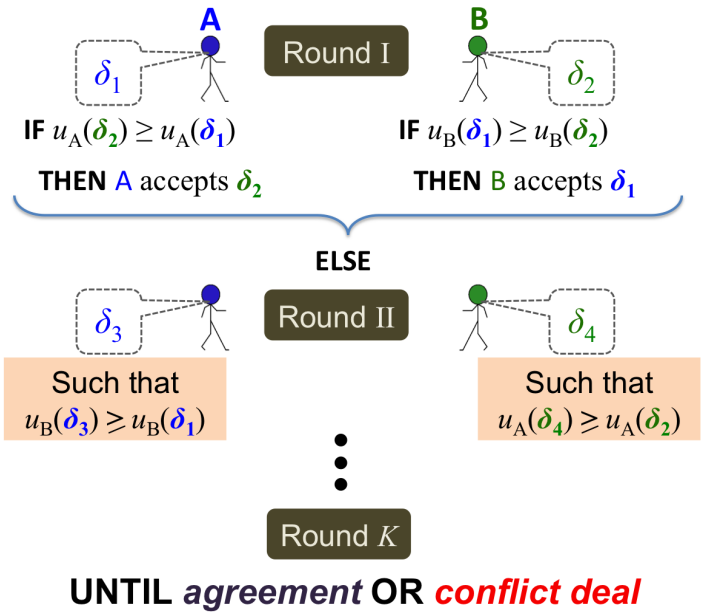
Task 1: Compute utilities Implement this computation of a utility of a deal for a specific agent. You may use the hard-coded costs of all subsets of tasks (addresses) provided with this assignment, but you need to implement a check whether a given δ is a valid redistribution (that is, whether $D_A \cup D_B = T$ and $D_A \cap D_B = \emptyset$ holds).

Task 2: Determine negotiation set See the definitions of *individual rationality* and *Pareto optimality* in the book or the slides. Determine for a given deal whether it is individually rational. Determine whether it is Pareto optimal. Finally, given a set of all possible tasks, determine the negotiation set.

3 Implementation of Monotonic Concession Protocol

See the MCP scheme on the next page, describing the protocol. In this part of the assignment you will implement this protocol and run a simulation with two agents.

Task 3: Propose a deal from the negotiation set Determine what the deal is that should be proposed, following the rules of the protocol. Check whether an agent should concede (or not, i.e., stay with the same deal) and if so, which deal shifts the balance to the other agent. Communicate this deal to the other agent.



Task 4: Test whether agreement is reached When you receive a deal from the other agent, check whether an agreement can be reached, i.e., whether this deal is as good or better as the deal that you proposed. Note that different deals may have the same utility - you cannot just compare the contents of the deal!

Task 5: Extend and experiment If all works fine with the running example, try to extend the program. Rather than hard-coding the tasks and the deals, let the negotiation start with either agent communicating T_i to the other agent, and determine T based on the information given by the agents (you'd probably still want to use the hard-coded cost of any subset of T). Experiment with different tasks and utilities, for example, changing the amount of addresses and/or the way utilities are computed. What are the effects of having more deals in the negotiation set?

4 Report your results

In your report you should evaluate your program by using the given running example. Give the content and evaluation of the deals that are proposed by both agents (see the MCP scheme). Then describe the effects of your extensions of the program (different addresses, more deals in the negotiation set, etc.). Try to interpret your findings: for example, what is the consequence (in complexity) of adding addresses?

5 For bonus points

Bonus exercise: Cheating Until now, both agents knew the original tasks allocation T_A and T_B (either because it was hard-coded or because they were communicated by the other agent), and this information was assumed to be correct. What if an agent pretends to have been allocated a non-existing letter, or hides a letter that does exist? If time permits, implement 'cheating' in the agent program. Demonstrate that either way of cheating can benefit the cheating agent. Report about the results.