TÉCNICO LISBOA

AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS

2019

EXERCISE AND ESSAY v1

Consider a world where agents are faced with task decisions in either single-agent or multi-agent settings.

Each task is seen as a lottery where the probability associated with a specific (sub)option is either given by a belief or implicitly defined based on the number of times an agent observed a given option.

Example: `(T1=[A=(60%,3),B=(40%,-1)],`
`T2=[A=(30%,[A1=(18,4),A2=(2,2)],B=(40%,[B1=(80%,2),B2=(20%,3)],C=(30%,1)])`

An agent can either select a single task or allocate a percentage of effort across the possible tasks, receiving in this latter case the proportional amount of utility. An agent can access the outcome of a decision when pursuing a single task, and the rounded, total utility when considering a mix of tasks.

Example:
```
agent> T2
(3,A1)
agent> (31%,T1;69%,T2)
(2,NA)
```

Encounters between two agents can occur. In this context, the perceived utility of executing a given task is conditional to the task being executed by the other agent. Agents may have a different set of elicitable tasks. Similarly to tasks in a single-agent setting, they are here seen as lottery based on believes and evidence.

Example: knowing a peer agent can execute tasks T1, T2 and T3, the aforementioned lottery is now given by:
`(T1|T1=…,T1|T3=…,T1|T4=…,T2|T1=…,T2|T3=…,T2|T4=…,)`

1. EXERCISE

1.1 SINGLE-AGENT DECISION [25%]

a. **[15%]** Program a rational agent that, given a lottery and a desirable number of decisions, places a decision in accordance with the introduced rules. Assume agent's rationality is oriented towards *current* utility (the utility after the decision) and can use the decision's outcome to improve behavior.

```
Example:
console> decide-rational (T1=[1,1],T2=[A1=(60%,-1),A2=(40%,1)]) 4
agent> T1
console> -3
agent> T2
console> -2
agent> T1
console> 1
```

At the end, the internal lottery for task 2 is `T2=[A1=(1,-2)]` since a single observation was recorded.

b. **[10%]** Program a risk-averse agent that, based on current observations, wants to maximize current utility while: 1) avoiding decisions with negative utility, and 2) distributing probabilities in the presence of selected tasks with equal utility expectations.

```
Example:
console> decide-risk (T1=[A1=(2,1)],T2=[B1=(40%,-1),B2=(60%,5)],T3=[C1=(1,2)])
agent> (2/3,T2;1/3,T3)
console> 1
```

1.2 MULTI-AGENT DECISION [40%]

Two competing agents may interact and the perceived expected conditional utilities seen as a game encounter given by a bi-matrix of payoffs per agent.

Assume agents are rational and:

- agents will opt for pure-strategies whenever there are Nash equilibria
  - if there is more than one Nash equilibrium, the Nash with the highest sum of payoffs should be returned. If still undecidable the Nash corresponding to the task with lower index for your agent and then for the peer agent should be returned
  - the observed outcome is used to update their beliefs
- agents will opt for mixed-strategies otherwise

```
Example:
// input line always separated by white-spaces (no newlines or tabs)
console> decide-conditional
          mine=(T1|T1=[A11=(1,1)],T1|T3=[A13=(1,-1)],T2|T1=[B21=(1,-1)],T2|T3=[B23=(1,1)])
          peer=(T1|T1=[A11=(1,1)],T1|T2=[A12=(1,1)],T3|T1=[B21=(1,-1)],T3|T2=[B23=(1,-1)])
          2
agent> mine=T2,peer=T1
console> mine=-3,peer=3
// updating beliefs leads to mine=(…T2|T1=[A211=(1,-1), A212=(1,3)]…)
// and other=(…T1|T2= [A121=(1,1),A122=(1,3)]…) leading to a bi-matrix without Nash equilibria
agent> mine=(50%,50%),peer=(50%,50%)
console> mine=1,peer=2
```

We can constrain the agent to uniquely search for Nash equilibria or for a mixed-strategy by replacing **`decide-conditional`** by either **`decide-nash`** or **`decide-mixed`**, respectively.
If there is no Nash or if task allocation (under a mixed-strategy) cannot be found the agent should return `blank-decision`

Program this agent behavior when facing a decision in the presence of a peer agent.
Note 1: guarantee the behavior of the agents allows decisions with more than 2 tasks
Note 2: you are **not allowed** to use libraries that calculate Nash equilibria: this results in a 0 grade
Note 3: you are **allowed** to use of algebra equation solvers to answer mixed strategies with >2 actions/tasks

2. ESSAY [35%]

In real-world scenarios, deception mechanisms can occur in human interactions, including:

a. *hidden* utilities: not sharing your beliefs on the expected utility of some actions/tasks;

b. *hidden* actions: pretending not being able to perform certain actions/tasks;

c. *phantom* and *decoy* actions: pretending you can realize certain actions/tasks you cannot.

Assuming the scenario described in 1.2, these notions can be similarly incorporated in agent interactions described by a game encounter. Accordingly, please write an essay with a maximum of **1.5 pages** (3 columns according to the AASMA template http://www.aamas2017.org/submission-instructions_aamas2017.php including all references) covering the following aspects:

- *specify* how each of the three listed deception mechanisms can be incorporated in a game encounter
- in accordance with the provided specification *discuss*:
    - how an agent can use deception to maximize his own reward
    - whether and how an agent can use deception in a cooperative setting
    - the impact of deception on Pareto optimality and Nash equilibrium of decisions

It is neither necessary nor suggested the search for state-of-the-art literature on this subject. Nevertheless, strong scientific statements should be backed-up by appropriate references it their evidence is not trivial.

This essay does not aim at being a survey. We suggest students to ponder on the use of these deception mechanisms in game encounters, and to extract principles to design rational agent behavior under both self-interested and cooperative contexts in accordance with their personal view.

The **clarity**, **adequacy**, **completeness**, **succinctness**, and **soundness** of the suggested agent behavior will be the major evaluation criteria of the essay.

3. SUBMISSION INSTRUCTIONS

Deadline: Thursday, April 11th

Mooshak platform will be used for code submission (single file) and **Fenix** for submitting the essay.

- exercises must be submitted to the Mooshak system; there is no other way of submitting the code.
- essays must be submitted to Fenix in .PDF format; there is no other form of submitting the essay.

Information on possible programming languages is available on the Mooshak system website.

The source code will be automatically evaluated by the Mooshak system.

Only the last submission will be considered for evaluation purposes. All previous submissions will be ignored.

Note that it is also possible to submit several times in Fenix, only the last being considered.

We encourage students to submit preliminary solutions to Mooshak and Fenix as early as possible.

Late deliveries will not be accepted.

The exercises' code and essay will be subjected to strict plagiarism checkers: if plagiarism is detect after clearance, the registration in AASMA is nullified and IST guidelines will apply.

Note 1: all the necessary contents to solve the exercise and write the essay will be lectured with an antecedence of over two weeks of the delivery/deadline. Nevertheless, eager students can read Chapter 11 of the book to easily access all the necessary foundations to answer the exercise and essay.

Note 2: consult FAQ on the course's webpage before posting questions to faculty hosts