MULTIPLE AGENT SYSTEMS - LECTURE 2

TYPED BY OHAD LUTZKY

1. Competitive Safety Analysis

Definition 1.1. A strategy $e \in S_i$ *dominates* a strategy $f \in S_i$ if for every $(s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n) \in \prod_{j \neq i} \Delta(S_j)$, we have

$$u_i(s_1, \ldots, s_{i-1}, e, s_{i+1}, s_n) \geq u_1(s_1, \ldots, s_{i-1}, f, s_{i+1}, \ldots, s_n)$$

with strict inequality for at least one tuple.

Essentially, e dominates f if the outcome by selecting e is better for player i, no matter what the other players' selections will be.

Definition 1.2. A game is called *non-reducible* if there do not exist $e, f \in S_i$ for some $i \in N$ such that e dominates i.

Definition 1.3. A game is called *generic* if for every pair of strategies $e, f \in S_i$ and $(s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n) \in \prod_{j \neq i} \Delta(S_j)$ we have that

$$u_i(s_i, \ldots, s_{i-1}, e, s_{i+1}, \ldots, s_n) = u_i(s_i, \ldots, s_{i-1}, f, s_{i+1}, \ldots, s_n)$$

iff $e = f$.

Essentially, this means that different strategies produce different outcomes.

Claim 1.4. Let G be a 2×2 non-reducible generic game. Assume that the optimal safety-level is obtained by a strictly mixed strategy. Then this value coincides with the expected payoff of that player in a nash equilibrium of G.

Example 1.5.

$$\begin{pmatrix} 100, 100 & 40, 210 \\ 60, 200 & 50, 90 \end{pmatrix}$$

This game has no pure equilibrium. There is a mixed one though: Agent 1 (rows) plays each strategy with probability 0.5, and agent 2 plays column 1 with probability 0.2.

A safety strategy, which will be Row 2 (not mixed), will give the row agent 50, whereas the equilibrium will get him 52.

Theorem 1.6. *There exists a 9/8-competitive safety strategy for the extended decentralized load balancing setting.*

Reminder: We have two data pipes. One, e_1, which works at rate 1, and the other, e_2, at rate $.5 < \alpha < 1$. Sending x data over e_1 gives us $\frac{x \cdot 1}{k}$, k being the number of players who selected the first pipe. Sending over e_2 will give $\frac{x\alpha}{l}$, l being the number of players who selected the second pipe.

Proof. Consider wlog $i = 1$. Consider the following equilibrium: $\left\{1, 2, \ldots, \left\lceil \frac{1}{1-\alpha} n \right\rceil\right\}$ will choose e_1, and the rest will choose e_2. Agent i's payoff is bounded by

$$\frac{x(1+\alpha)}{n} \tag{1}$$

Now consider the following strategy: Choose e_1 with probability $\frac{\alpha}{1+\alpha}$ (notice that this is not part of a nash equilibrium - it's between $\frac{1}{3}$ and $\frac{1}{2}$). Assume n participants, where $\beta(n-1)$ of the other participants (excluding i) use e_2 ($0 \le \beta \le 1$). The expected payoff of i:

$$\begin{aligned}
&\underbrace{\frac{1}{1+\alpha}}_{e_2 \text{ probability}} \cdot \frac{\alpha x}{\underbrace{\beta(n-1)}_{\text{Other people}} + \underbrace{1}_{\text{Me}}} + \frac{\alpha}{1+\alpha} \cdot \frac{x}{(1-\beta)(n-1)+1} \\
\ge& \frac{1}{1+\alpha} \frac{\alpha x}{\beta n + 1} + \frac{\alpha}{1+\alpha} \frac{x}{(1-\beta)n+1} \\
=& \frac{x\alpha}{1+\alpha} \frac{n+2}{(1+\beta n)(n-\beta n+1)} \qquad (2)
\end{aligned}$$

To get our c ratio, we'll divide (1) by (2), and we'll get

$$\frac{(1+\alpha)^2}{\alpha} \cdot \frac{(\beta-\beta^2)n^2 + n + 1}{n(n+2)}$$

When $n \to \infty$, we get $\frac{(1+\alpha)^2}{\alpha}\left(\beta - \beta^2\right)$. Taking $0 \le \beta < 1$, $0.5 \le \alpha < 1$, and get the $\frac{9}{8}$ bound. That is, the nash equilibrium is at most $\frac{9}{8}$ times better than our safety-level strategy. □

We will move on to a world in a which a planner can advise me on what to do, for example - conventions. As a matter of fact, here it's still not clear whether the agents will follow the advice they get, unless - for example - everyone has similar preferences.

In game theory, we often get multi-stage games, which are extremely complex to describe with the standard tables we used so far. The strategy we choose in each step will be a function of the history of all players' actions. It also depends on the number of iterations we have in the game: For example, in a finitely repeated prisoner's dillema, because of backward induction, defecting is the best option. In an infinite one, Tit-for-Tat seems to be the best. A great paradox occurs in the centipede game, where backwards induction leads us to the conclusion that *rational* players must defect immediately. Thus we show a great problem, from the perspective of modeling multi-agent systems, with the realism of the concept of a game.

We will now move on to speak about a situation where a designer sends out agents to play a game. Obviously, it will not be possible to specify rules and conventions for every situation we may encounter, therefore we will attempt to design 'Social Norms'. Later on, we will move on to systems where we do not control all of the agents.

2. Multi-Entity Models

Definition 2.1. A *multi-entity model* is a tuple $(E_1, E_2, \ldots, E_n, \mathcal{A}, T)$.

E_i**:** Agent i

$\mathcal{A}$**:** Set of actions
T**:** A state transition function

E_i itself is a finite state machine (L_i, A_i):
L_i**:** set of local states
$A_i : L_i \to 2^A$**:** determines possible actions at each local state

$\mathcal{A}$, *system configuration*, is a tuple $(s_1, \ldots, s_n)$ of the states of the different agents.

C is the set of system configurations.

Λ - a distinguished action (no action). Passive agents perform only Λ.

The tuple of actions $(a_1, \ldots, a_n)$ executed by the agents of a particular point is called a *joint action.*

$$T : C \times \mathcal{A}^n \to C$$

This is a very general observation, which we will try to cut down for more useful analysis.

Every agent i is comprised of a set of components: $\{M_{i_1}, M_{i_2}, \ldots, M_{i_{f(i)}}\}$. The state s_i of agent i is comprised of the states of the components:

$$s_i = \left(l_{i_1}, l_{i_2}, \ldots, l_{i_{f(i)}}\right)$$

The state space of M_{i_j} will be called L_{i_j}, and we must demand that L_{i_j} be pairwise disjoint.

2.1. **Efficient representation of the transition function.** T is replaced by a collection of T_{i_j}.

$$T_{i_j} : L_{i_j} \times \mathcal{A}^n \times \mathcal{L} \to L_{i_j}$$

$\mathcal{L}$ being a set of conditions on the system. $\mathcal{L}$ is a propositional language, closed under the boolean operators $\wedge, \neg$, whose primitive elements are of the form $In(s)$ for all states s such that $s \in L_{i_j}$ for some M_{i_j}.

A default: $T_{i_j}(s, \vec{a}, \psi) = s$ for all ψ and s.

Popular representation: Transition from a certain state x to a certain state y of the machine M_{i_j}, under some joint action, is performed under a certain condition in $\mathcal{L}$.

Usually one adds an efficient representation for the A_is: $A_i : \mathcal{A} \to \mathcal{L}_i$. That is, for each action, what are the conditions for performing it - $\mathcal{L}_i$ being $\mathcal{L}$ limited to the states of agent i.

Finally, if there is an action which does not affect s, we will not mark it.

What we're missing is goals. A goal can be a logical formula in $\mathcal{L}$.

Assume that the system start in an initial configuration c_0 which is taken from a set of possible initial configurations $C_0 \subseteq C$.

Every agent can see, at any point, his own state.

A plan/strategy for an agent is a function from the history of his observations (states he's visited) and actions he's performed to his next action.

Now we have a multi-entity model, in which every agent has his own goals, and we're trying to give every agent a plan such that all agents will attain their goals (at some point or simultaneously) *regardless of the initial configuration*, as long as it's in C_0.

This problem is called *Cooperative Goal Achievement.* One may ask whether it is possible to give the agents finite-length plans for attainment of the goals.