Strategic Interaction Definition Language

Rustam Tagiew (tagiew@informatik.tu-freiberg.de)
Institute for Computer Science of TU Bergakademie Freiberg, Germany

Games are used as synonym to strategic interactions. A finite game has a finite number of participants, of their actions and of states. According to Nash, a finite game has at least one (mixed strategies) equilibrium. Finding the equilibria is one of the objectives in game theory. In Artificial Intelligence, games are interesting from at least two points of view - mechanism design and agent design. In mechanism design, one wants to achieve a kind of (mostly cooperative) behavior and needs rules. For agent design, one has rules and tries to derive optimal behavior. In most cases, the system represents one special game like chess and has to compute the best strategy as quickly as possible. In such cases, the game is represented by low level code. In practical view, every agent has to represent the game and then a component is required which represents the game for all agents. In global view, this required component is a (game) server, which implements rules for interaction. This means, that one has to solve in general two tasks for game computing - server and solver. Game solver is the part used by agents. There are many techniques for solving games - from optimal game theoretic to suboptimal AI heuristics. If one has a library with code for a chess game server and a library for an agent, which does his best in playing chess, it does not mean that one can use the libraries for other games. In some cases, there is additionally a redundant game representing code on both sides. The low-level code based game server can not send the game rules to the agents, unless he sends his code. The game rules are typically reflected in a network communication protocol for the server. Consequently, if an agent does not have a representation of the game, he can not participate properly in it. And one can not describe in a clean way a behavior of an agent as depending on rules of the game, because these rules are hard-coded. To get rid of these problems, one has to develop a high level language to define games. Summarized, it causes the following advantages. (1) A clear and convertible representation of a game is given. Convertible means that it can be easily converted to other representation formats and formalisms. Such representations can be exchanged between systems - e.g. from a game server to a client. (2) The game solver and the game server use the same file. This averts redundancy. It also guarantees that both run on the same game. (3) The rules can be edited easier. One can also develop algorithms, which manipulate game representations in any possible way - e.g. 'remove simultaneous turns' or 'reduce the number of agents'. It can be used for evolutionary mechanism design, where the game description must be able to mutate. (4) One can develop techniques for defining behaviors of agents on the basis of a game description. Players which are defined for playing multiple kinds of games can be constructed and benchmarked. The only disadvantage can be the computation time. Low level code solutions for specific games can be done smarter. That saves computation time.