Engineering Applications of Artificial Intelligence 59 (2017) 148–158

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Co-Utility: Self-Enforcing protocols for the mutual benefit of participants

Josep Domingo-Ferrer, Sergio Martínez, David Sánchez*, Jordi Soria-Comas

UNESCO Chair in Data Privacy, Department of Computer Science and Mathematics Universitat Rovira i Virgili, Av. Països Catalans 26, E-43007 Tarragona, Catalonia, Spain

ARTICLE INFO

Keywords:
Agents
Protocols
P2P
Self-enforcement
Game theory

ABSTRACT

Protocols govern the interactions between agents, both in the information society and in the society at large. Protocols based on mutually beneficial cooperation are especially interesting because they improve the societal welfare and no central authority is needed to enforce them (which eliminates a single point of failure and possible bottlenecks). In order to guide the design of such protocols, we introduce co-utility as a framework for cooperation between rational agents such that the best strategy for each agent is to help another agent achieve her best outcome. Specifically, in this work we study and characterize self-enforcing protocols in game-theoretic terms. Then, we use this characterization to develop the concept of co-utile protocol and study under which circumstances co-utility arises. Furthermore, we give a detailed study of co-utile protocol design in the case of anonymous query submission to a web search engine. The theoretical analysis is complemented with empirical results obtained from an implementation in a simulated multi-agent environment, which illustrates how co-utility can make cooperation self-enforcing and improve the agents' welfare.

1. Introduction

A protocol specifies a precise set of rules that govern the interaction between agents performing a certain task; that is, it details the expected behavior of each agent involved in the interaction for the task to be successfully completed. For example, for vehicles to avoid collisions at an intersection, they must wait for the green light. Also, in information technology there are plenty of protocols: the Internet Protocol (IP) defines how to route information packets in the Internet, the MESI protocol (Papamarcos and Patel, 1984) defines how to preserve coherence between cache memories in multiprocessor architectures sharing a main memory, etc.

For protocols to be effective, they must be adhered to. This is not problematic when the participating agent cannot deviate by design, such as in the MESI protocol, but it becomes an issue when the agents are free to choose between following the protocol or not, as it happens with vehicles at a crossing regulated by traffic lights. Although free agents cannot be forced to follow a protocol, rational free agents can be persuaded to do so if the protocol is properly designed. Such properly designed protocols will be called self-enforcing in the sequel. Examples of self-enforcing protocols that can be found in the literature include those involved in rational multiparty computation (Dodis and Rabin, 2007), the shotgun clause (Brooks et al., 2010) (which is a way for two rational agents to agree on the price of an item) and the Vickrey auction (Vickrey, 1961) (which is a kind of auction in which each rational agent truthfully reports her valuation), among others (Schneier, 2016).

While self-enforcement is essential, we are interested in protocols offering more than that: we want self-enforcing protocols that result in mutual help between agents and we call them *co-utile protocols*. A prominent advantage of co-utile protocols is that they promote social welfare. To illustrate, consider an agent that is interested in querying a web search engine but does not want the search engine to learn her queries, because these may disclose her personal features or preferences. If there is another agent also interested in privacy-aware querying, both agents can exchange (some of) their queries (and results), thereby preventing the web search engine from accurately profiling either agent; this results in a mutually beneficial collaboration (Domingo-Ferrer and González-Nicolás, 2012a, 2012b).

Co-utile protocols can be crafted for scenarios where the interests of the agents are complementary –or can be made complementary by adding appropriate incentives–, so that helping other agents becomes the best way of pursuing one's own interests. Similar ideas about adding artificial incentives to promote cooperation have been proposed whose scope is narrower and more ad hoc than the one of the co-utility framework developed in this paper. For example, in P2P networks for sharing of distributed resources (e.g., storage, computing, data, etc.), incentives are used to achieve self-enforcing collaboration and deter the so-called *free-riders* (that is, peers who use resources from others but who do not offer their own resources) (Rahman et al., 2011); incentives in this context take the form of better service (Buragohain

* Correspondence to: Departament d'Enginyeria Informàtica i Matemàtiques, Universitat Rovira i Virgili, Av. Països Catalans 26, 43007 Tarragona, Catalonia, Spain.
E-mail address: david.sanchez@urv.cat (D. Sánchez).

http://dx.doi.org/10.1016/j.engappai.2016.12.023
Received 21 January 2016; Received in revised form 3 November 2016; Accepted 27 December 2016
Available online 05 January 2017
0952-1976/© 2016 Elsevier Ltd. All rights reserved.

et al., 2003), or some sort of virtual money (Friedman et al., 2006) for those who contribute.

In this paper, we first formalize the notions of protocol and self-enforcing protocol. Then we move on to define co-utile protocols. Since we are assuming rational agents who freely decide whether to adhere to the protocol or not, game-theoretic modeling arises as the most natural choice. The assumption of free rational agents is plausible in peer-to-peer (P2P) scenarios lacking a central authority and a common legal framework that can be used to enforce a specific behavior. The power of co-utility is illustrated in a case study that deals with P2P anonymous query submission to a web search engine or a database. In this context, we present a set of self-enforcing and mutually beneficial protocols.

The remaining sections are organized as follows. In Section 2 we formalize the concepts of protocol and self-enforcing protocol. In Section 3 we introduce and discuss the notions of co-utility and co-utile protocol. In Section 4 we apply the previous concepts to the anonymous query submission case study and provide a set of co-utile self-enforcing protocols. In Section 5 we empirically test the designed protocols in a simulated multi-agent system. We conclude in Section 6 and sketch some future research lines.

2. Self-enforcing protocols

Since this paper focuses on protocols, we first need to clarify what we understand by *protocol*. Loosely speaking, a protocol is a sequence of interactions among a community of agents, called steps, that are aimed at carrying out a certain task.

A formalization of the concept of protocol that is often used in computer science is based on *finite state machines*. A finite state machine is a mathematical model of computation. It consists of a set of states, one of which is the current state, and a set of transitions between states that are triggered by specific events and modify the current state. While a finite state machine nicely models a protocol (each step changes from one state to another), it fails to capture the behavior of rational agents who choose their actions with the aim of maximizing their utility.

With rational agents in mind, game theory (Leyton-Brown and Shoham, 2008; Osborne and Rubinstein, 1994) seems the right mathematical model. This theory models interactions between self-interested agents that act strategically. An agent is self-interested if she defines a preference relation over the set of possible outcomes of the protocol. On the other hand, an agent acts strategically when she takes into account her knowledge and expectations about the state of the world and about other agents to decide on her strategy (the way she plays the game). Game theory identifies subsets of outcomes (a.k.a. solution concepts) that agents would be most interested in achieving. In this work the focus will be on equilibrium solutions, i.e., outcomes which rational agents have no motivation to deviate from.

In our proposed formalization, a game is used to model all the possible interactions among agents in the underlying scenario. In particular, the game includes also those interactions among agents that are not desired. Then, a protocol is regarded as a prescription of a specific behavior in the underlying scenario, that is, a sequence of desired interactions.

The type of interaction between agents is a key point in the outcome of a game. Game theory can model several interaction types, including:

- *Simultaneous and sequential moves*. Moves in a game are called simultaneous if each agent chooses her move independently (unaware) of the other agents' moves. On the other side, moves are called sequential if, at the time of choosing a move, previous moves made by other agents are known (at least to some extent).
- *Perfect and imperfect information*. A sequential game (one with sequential moves) is said to be a perfect-information game if the agent about to make her move has complete knowledge on the previous moves made by the other agents. If the agent's knowledge on previous moves is only partial, the game is said to be an imperfect-information game.
- *Complete and incomplete information*. If the previous category referred to knowledge on previous moves, this category refers to agents' knowledge on the underlying game. In games with complete information, the payoff of each agent at each final state is known by all agents. In games with incomplete information (a.k.a. Bayesian games), an agent is uncertain about the payoffs of the other agents.

The actual formalization of a game depends on the type of interaction one wants to model. In this paper we focus on sequential games with perfect information (Kuhn, 1997), because this is a quite common and basic type of interaction between agents. Other scenarios involving uncertainties about the game are certainly conceivable (Buccafurri et al., 2008), but we leave for future work the generalization to arbitrary sequential games (with perfect or imperfect information).

The formal definition of a sequential game with perfect information is as follows:

Definition 1. *(Perfect-information game)*A perfect-information game (in extensive form) is a tuple $G = (N, A, H, Z, \chi, \rho, \sigma, u)$, where:

- N is a set of n agents;
- A is a set of actions;
- H is a set of non-terminal choice nodes;
- Z is a set of terminal nodes, disjoint from H;
- χ: $H \to 2^A$ assigns to each choice node a set of possible actions;
- ρ: $H \to N$ assigns to each choice node a player $i \in N$ who chooses an action at that node.
- σ: $H \times A \to H \cup Z$ is an injective map that, given a pair formed by a choice node and an action, assigns to it a new choice node or a terminal node;
- $u = (u_1, \ldots, u_n)$, where u_i: $Z \to \mathbb{R}$ is a real-valued utility function for agent i on the terminal nodes.

A perfect-information game can be represented in the so-called *extensive form* as a tree where:

- Each non-terminal choice node is labeled with the name of the agent making the move;
- Each terminal node is labeled with the utility that each agent obtains when reaching it;
- Edges going out from a node represent the actions available to the agent making the move.

Although the focus has been placed on sequential games with perfect information, nothing has been said so far about the completeness of the information. Assuming complete information seems too restrictive when trying to model the interactions between a set of potentially unrelated agents. Fortunately, the above tree representation can easily accommodate both complete and incomplete-information games:

- In games with complete information, the utilities at the terminal nodes are fixed values known to all the agents. Fig. 1 shows sequential adaptations of two well-known games: the Prisoners' Dilemma and the Battle of the Sexes (Luce and Raiffa, 1957). Both are perfect-information and complete-information games.
- In games with incomplete information, the utilities at terminal nodes are not completely known. This is modeled by replacing the fixed utilities at terminal nodes by utility functions that depend on an additional parameter: the type of each agent. The type of an agent encapsulates all the information on that agent that is not common knowledge. The set of types of the game is the Cartesian product of the set of types of each agent: $\Sigma = \Sigma_1 \times \ldots \times \Sigma_n$. Each agent knows her type but is uncertain about the types of the other agents. The agent models this uncertainty by attributing to every other agent a prior

Download English Version:

https://daneshyari.com/en/article/4942810

Download Persian Version:

https://daneshyari.com/article/4942810

Daneshyari.com