Co-Utility: Self-Enforcing Protocols for the Mutual Benefit of Participants

Josep Domingo-Ferrer, Sergio Martínez, David Sánchez[1], 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

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.

Keywords: Agents, Protocols, P2P, Self-enforcement, Game theory

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 [25] defines how to preserve coherence between cache memories in multiprocessor architectures sharing a main memory, etc.

[1] Corresponding author. Address: Departament d'Enginyeria Informàtica i Matemàtiques, Universitat Rovira i Virgili, Av. Països Catalans 26, 43007 Tarragona, Catalonia. Tel.:+34 977 559657; Fax: +34 977 559710. E-mail: david.sanchez@urv.cat

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 [8], the shotgun clause [2] (which is a way for two rational agents to agree on the price of an item) and the Vickrey auction [31] (which is a kind of auction in which each rational agent truthfully reports her valuation), among others [30].

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 [10, 11].

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) [27]; incentives in this context take the form of better service [5], or some sort of virtual money [13] 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 [17, 24] 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 [16], because this is a quite common and basic type of interaction between agents. Other scenarios involving uncertainties about the game are certainly conceivable [3, 4], 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;

- $\chi : H \to 2^A$ assigns to each choice node a set of possible actions;

- $\rho : H \to N$ assigns to each choice node a player $i \in N$ who chooses an action at that node.

- $\sigma : H \times A \to H \cup Z$ 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. Figure 1 shows sequential adaptations of two well-known games: the Prisoners' Dilemma and the Battle of the Sexes [18]. 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 distribution over the possible types. Usually the same prior distribution is assumed for all agents.

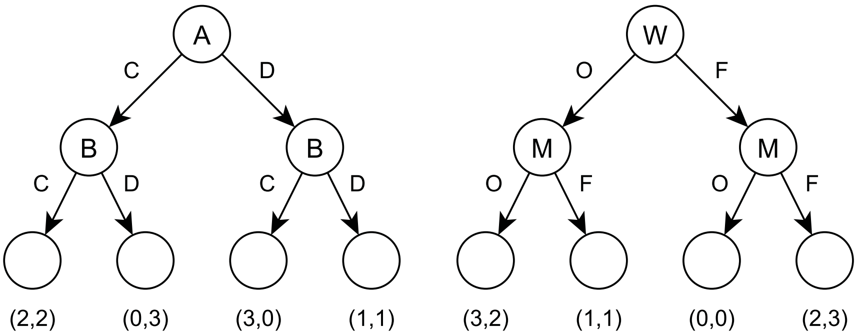

Figure 1: Sequential versions of two well-known simultaneous games: the Prisoners' Dilemma (left) and the Battle of the Sexes (right). In the first game, prisoner A first decides whether to cooperate (C) with prisoner B, by *not* betraying him, or defect (D), by betraying him; then prisoner B faces the same decision (unawares of A's move in the simultaneous version). The utility of each prisoner is the reduction of the number of years in prison she or he obtains. In the second game, the woman, W, in a couple chooses between going to the opera (O) or to a football match (F); then the man, M, does the same (unawares of W's choice in the simultaneous version). The utility is how much they enjoy themselves: both would like to be together, but W prefers opera and M football.

We have defined a game as containing the set of actions, A, that are available to the agents. The strategies of a game describe the possible ways in which each agent can choose actions. The most basic type of strategies are pure strategies, defined as follows.

Definition 2. *(Pure strategy)* Let $G = (N, A, H, Z, \chi, \rho, \sigma, u)$ be a perfect-information game in extensive form. A pure strategy for an agent i is a function s_i that selects an available action at each non-terminal node at which i has to make a decision, that is, at each $h \in H$ such that $\rho(h) = i$. We denote by S_i the strategy set of agent i, that is, the set of pure strategies available to agent i.

In other words, a pure strategy provides a complete definition of how an agent will play a game, because it determines the move the agent will make for any situation he or she could face. Beyond pure strategies, more sophisticated strategies can be defined, such as mixed and correlated strategies, in which the selection of actions at each decision node h is randomized. We can now define the notion of strategy profile.

Definition 3. *(Strategy profile)* In a game G with n agents, a strategy profile s is a tuple $s = (s_1, \dots, s_n)$, where s_i is the strategy chosen by agent i. The set of possible strategy profiles is $S_1 \times \dots \times S_n$, where S_i is the set of strategies available to agent i.

As introduced above, after modeling the underlying scenario as a game, we can now model a protocol as a sequence of prescribed interactions. That is, a protocol embodies the agents' behavior that the designer wants to favor.

Definition 4. *(Protocol).* Given a perfect-information game G, a protocol P is a set of strategy profiles. If, at each non-terminal node h, agent $\rho(h)$ is allowed only one action by the protocol, then the protocol consists of a single strategy profile.

If G is represented in extensive form as a tree, a protocol can be viewed as a subtree from the root to several leaves. If the protocol consists of a single strategy profile, then it can be viewed as a path from the root to a leaf.

We are interested in self-enforcing protocols, that is, in prescribing behaviors that are adhered to by rational agents. In other words, no rational agent should have any incentive to deviate from the protocol provided that the other agents stick to it. This requirement can be rephrased in game-theoretic jargon by saying that each of the strategy profiles in the protocol must be an equilibrium of the underlying game. There is a large variety of equilibrium concepts such as Nash equilibrium, correlated equilibrium, subgame-perfect equilibrium, etc. In the context of protocols defined over perfect-information sequential games, we expect each agent to select a move that yields an equilibrium of the remaining subgame, that is, of the portion of the game that remains to be played after some moves have already been made by the agents. This kind of equilibrium is known as subgame-perfect equilibrium.

Definition 5. *(Subgame).* Given a game G in extensive form, the subgame of G rooted at node h is the restriction of G to h and its descendants.

Definition 6. *(Subgame-perfect equilibrium).* Given a game G in extensive form, a strategy profile s is a subgame-perfect equilibrium if, for any subgame G' of G, the restriction of s to G' is an equilibrium.

When computing Nash equilibria to check whether Definition 6 holds for them, in the case of complete-information games the given utilities are used, whereas in the case of incomplete-information games one resorts to expected utilities (derived from the prior distribution of types).

We are now in a position to formally define self-enforcing protocols using the concept of equilibrium:

Definition 7. *(Self-enforcing protocol).* A protocol P over a perfect-information game G is self-enforcing if no agent can increase her utility by deviating from one of the strategy profiles in P, provided that the other agents stick to that strategy profile. Equivalently, each of the strategy profiles in P is a subgame-perfect equilibrium of G.

In the sequential Battle of the Sexes (BoS) game described in Figure 1, the protocol $P = (O, O)$ is self-enforcing: (O, O) is a Nash equilibrium of the BoS game, and (O) is a Nash equilibrium of the game that remains after the first action.

An interesting property of subgame-perfect equilibria is that they always exist; that is, every perfect-information game has a subgame-perfect equilibrium. Such an equilibrium is usually found by means of backward induction [17]. The backward induction algorithm assumes that, at each node, the agent making the move selects the action that gives her the best outcome. The algorithm starts evaluating the choice nodes that are parents of terminal nodes. Once the moves at these choice nodes have been selected, the algorithm proceeds backwards by evaluating the choice nodes that are parents of the previous choice nodes. For example, in the sequential BoS game (see Figure 1), M is the agent making the last choice. In the left branch, the best option for M is O, which leads to utility $(3, 2)$. In the right branch, the best option for M is F, which leads to utility $(2, 3)$. Because W knows that M will seek to maximize his own utility, W can simplify the original tree to get a tree with a single choice node (hers) in which choosing O leads to utility $(3, 2)$ and choosing F leads to utility $(2, 3)$. Thus, in the original (unsimplified) tree with two decision stages, W should select O in the first place, and M should then also select O.

Let us find self-enforcing protocols in the two games proposed in Figure 1. In the sequential version of the Prisoners' Dilemma, it is easy to see that the only self-enforcing protocol is (D, D). Choosing C is not self-enforcing for any agent because switching from C to D improves the agent's utility (provided that the other agent does not alter his/her move). In the sequential BoS game, there are two self-enforcing protocols: (O, O) and (F, F). In either protocol, no agent can improve her utility by deviating from the protocol provided that the other agent sticks to it. Of course, since in the sequential version agent W chooses her move first, she is most likely to favor (O, O) over (F, F).

3. Co-Utility and Co-Utile Protocols

As motivated in the introduction, we are interested in protocols that, beyond being self-enforcing, induce a rational agent to collaborate with other agents to increase their utilities. We say a protocol promotes collaboration between a set of agents if the utility of each participating agent is strictly greater than her utility when not participating (an agent is said to participate in the protocol if and only if she follows the prescriptions of the protocol). More precisely, we are interested in a protocol that promotes the most rewarding collaboration for all agents, that is, such that there is no alternative protocol whereby all agents could get a better outcome and at least one of them a strictly better outcome; in game-theoretic terms, we seek Pareto-optimality of the utilities of agents. This Pareto-optimal collaboration is the essence of co-utility, which can be defined as follows:

Definition 8. *(Co-utile protocol).* A self-enforcing protocol P is co-utile if each strategy profile in P is Pareto-optimal.

In the sequential BoS game described in Figure 1, the protocol $P = (O, O)$ is co-utile. As we have already seen that it is self-enforcing, it only remains to check that it is Pareto optimal: W cannot improve her utility with respect to (O, O); M increases his utility if (F, F) is played, but that decreases the utility of W.

Co-utility is reminiscent of cooperative game theory [7] in that it deals with agents that collaborate to get a more desirable outcome. However, there is a substantial difference: in cooperative game theory, coalitions of agents are formed and the agents in a coalition coordinate their strategy to maximize the coalition's payoff, which is then divided among the agents. Assuming that the payoff can be divided among the agents in a coalition is essential for the coalitions to be formed. In contrast, in co-utility we assume that each agent acts autonomously and keeps to herself the payoff she gets: this allows co-utility to deal with non-divisible payoffs, such as privacy or security.

On the other hand, since the goal of co-utility is to lead to (co-utile) protocols, mechanism design is a natural framework to compare with. In mechanism design [26, 19], the ultimate goal is to come up with mechanisms that yield a previously defined socially desirable outcome. In this sense, co-utility is less demanding than mechanism design: it does not aim at enforcing a specific socially beneficial outcome, but rather at promoting a mutually beneficial collaboration between agents. It may well happen that the mutually beneficial outcome of a co-utile protocol is also socially desirable (but this does not necessarily happen).

Another difference between mechanism design and co-utility is that the former requires preference alignment, whereas the latter does not:

- In mechanism design, the outcome is selected based on the preferences reported by the agents. Untruthful reporting of preferences by agents in an attempt to obtain a more desirable outcome is an important issue that must be addressed by mechanism design. This is usually tackled by requiring some payment from agents that is calculated to make misrepresentation of utilities unfruitful. This kind of mechanism to align the preferences of the individual agents with the socially desirable outcome is known as incentive-compatible mechanism.

- In co-utility, rather than aligning preferences to the socially desirable outcome via incentives, we seek to promote collaboration between agents that have complementary preferences. For instance, for the case of privacy-preserving query submission to a web search engine, we assume that the agents that participate have complementary preferences regarding the topics they wish to query about, but they are all concerned about their privacy; in Section 4 we exploit this fact and give a win-win solution whereby agents preserve their own privacy by helping other agents preserve theirs. Moreover, decisions in co-utility are made by the agents themselves rather than in a centralized way: when asked for collaboration, each agent decides which strategy is best for her.

In the sequential version of the Prisoners' Dilemma (Figure 1), the only self-enforcing protocol is (D, D) but it is not Pareto-optimal because (C, C) provides a strictly greater payoff to both agents. Therefore, in this game no co-utile protocol exists. In the sequential version of the Battle of the Sexes (Figure 1), it is easy to check that both self-enforcing protocols (O, O) and (F, F) are Pareto-optimal solutions of the game. Thus, both protocols are co-utile.

A specially interesting case of co-utility happens when the protocol is not only Pareto-optimal, but maximizes the utility of all agents:

Definition 9. *(Strict co-utility).* A protocol P on a game G is strictly co-utile if the utility that every agent derives from following any strategy in P is strictly greater than the utility the agent would obtain by following any strategy not in P.

We next prove that strict co-utility as per Definition 9 implies co-utility as per Definition 8.

Proposition 1. *If a protocol P on a game G is strictly co-utile, then it is co-utile.*

Proof. We must check that P is self-enforcing and Pareto-optimal. Since all strategies not in P yield less payoff than any strategy in P, we have that:

1. No agent can increase her payoff by deviating from P and, thus, P is self-enforcing;

2. No protocol P' different from P yields a better utility to all agents and a strictly better utility to at least one agent; hence, P is Pareto-optimal.

□

Co-utility is about making selfish behavior compatible with mutually beneficial collaboration. However, given a game G, in general there is no guarantee that the selfish behavior of the agents will lead to co-utility, let alone strict co-utility, even if there are co-utile (or strictly co-utile) protocols in the game. However, for some specific games, such a guarantee can be given:

Proposition 2. *In a perfect-information game G where all the agents maximize their utilities in exactly the same set of terminal nodes of the tree that represents the game in extensive form (and only in these terminal nodes), selfish behavior by the agents will cause them to follow a strictly co-utile protocol.*

Proof. Let $\mathcal{M}$ be the set of terminal nodes where the utilities of all agents are maximized. Let l be the length of the shortest path from the root to a node in $\mathcal{M}$. We will prove that selfish behavior leads to a node in $\mathcal{M}$ by induction over l. If $l = 1$, there is a single action to be chosen and at least one node $m \in \mathcal{M}$ can be reached by the agent making the choice. This agent could certainly take a longer path (with more than one action to be chosen), but this path should equally lead to a node in $\mathcal{M}$ (otherwise the agent would obtain a suboptimal payoff). We assume that the proposition is satisfied when $l \leq k$ and we need to show that it is also satisfied for $l = k+1$. To apply the induction hypothesis, we split the shortest path to $\mathcal{M}$ into two parts: P_1 (containing the k initial steps) and P_2 (containing the final step). The same reasoning used for $l = 1$ shows that the agent making the decision at $leading(P_2)$, the leading node of P_2, chooses to follow P_2. As we have determined the path that will be followed if $leading(P_2)$ is reached, we can simplify the game tree by removing all the subtrees rooted at $leading(P_2)$ and copying the utilities of m (which are the same as the utilities of any node in $\mathcal{M}$) to $leading(P_2)$. To show that m is reached in the original tree, it is enough to show that $leading(P_2)$ is reached in the simplified tree. But the latter is immediate by applying the induction hypothesis to P_1 , because the terminal node of P_1 is $leading(P_2)$. □

A co-utile protocol always prescribes selfish behaviors (because it is self-enforcing) but not all selfish behaviors need to be co-utile. The interest of Proposition 2 is that it describes a class of games in which selfish behavior by everyone always yields strict co-utility. This is indeed very helpful to find a co-utile protocol. The alternative to find co-utile protocols is to use the backward induction algorithm (see [17]) on the game tree in order to seek subgame-perfect equilibria. However, this algorithm involves substantial computational cost, because it needs to traverse all the nodes of the tree (from leaves to root).

4. Rational and Anonymous Query Submission

In order to illustrate the above theoretical concepts, we show in this section how to design co-utile protocols in a practical scenario.

Consider an agent who wants to submit queries to a web search engine (WSE) or a database without the latter learning her interests. That is, the agent wants to avoid being profiled by the WSE. The agent's defense consists of making her profile diverse enough so that the WSE cannot determine her interests. In essence, to mask her profile, the agent must hide the real queries she is interested in among a set of queries about diverse topics she is not interested in.

The agent can perform the hiding on her own, by generating random fake queries and submitting them along with her real queries [9, 14]. Such a standalone approach has several downsides: on the one hand, fake queries overload the WSE, so if every agent followed this approach, the WSE performance would strongly degrade; on the other hand, it is not so easy to produce fake queries that are plausibly indistinguishable from real ones [20].

Relying on a set of peer agents (as proposed in [28, 6]) is an alternative that is free from the previous downsides. Consider several agents with similar privacy interests. To mask her profile, an agent can ask another agent to submit her query to the WSE rather than submitting it herself. Also, the agent can submit to the WSE real queries originated by other agents. In this way, a rational cooperation between peer agents emerges [10, 11] and all of them manage to blur their profile of interests without having to submit fake queries. In this section, we analyze this cooperation under the co-utility framework.

The primary utility agents want is functionality: they want to get their queries answered. A secondary utility is privacy: they may also wish their interest profiles to stay private w.r.t. the WSE. Note that any WSE can be expected to try to profile its users (typically for marketing purposes) according to the topics mentioned in the queries they perform. We will call atomic utilities the two components of an agent's utility, namely functionality and privacy. These two atomic utilities are not easily reconciled, because an agent alone cannot optimize both of them simultaneously.

4.1. Single-hop query submission game with two agents

The first scenario/game we analyze, which we call *single-hop query submission*, is restricted to two agents: the initiator of the query (I) and the responder to a query submission request (R). The initiator I is interested in functionality (having her query answered) and privacy w.r.t. the web search engine (keeping her interests private). If the new query diversifies her interest profile (for example, because she had never asked a similar query), I is likely to submit it herself to the WSE; otherwise, I will ask R to submit the query on her behalf. The responder R is only concerned about his privacy (whether I's query is useful to blur R's profile w.r.t. the WSE).

The interaction among I, R and the WSE is as follows:

- I chooses between submitting the query to the WSE herself or forwarding the query to R for submission.
- Upon query reception, R chooses between submitting the query to the WSE (and returning the results to I) or declining to submit it.

These interactions are illustrated in Figure 2. Since (S) and (F, D) do not have any effect on the privacy or the utility of the responder, we assume that the responder's payoff in these protocols (u_S^R and u_{FD}^R, respectively) is 0. It

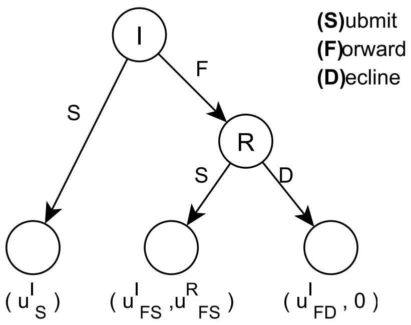

Figure 2: Tree showing the possible interactions in the single-hop anonymous query submission game between two agents: initiator (I) and responder (R). To keep the figure compact, we only represent the payoffs of the agents active in each interaction: for interaction (S), the payoff of any agent other than the initiator is 0; for the other interactions, only the payoffs of the initiator and the active responder are represented, the rest of agents' payoffs being 0.

is also reasonable to assume that $u^I_{FS} > u^I_{FD}$ as both (F, S) and (F, D) are equivalent in the initiator's eyes regarding privacy w.r.t. the WSE but (F, S) is better in terms of functionality.

Given the above assumptions about the payoffs of the agents under different protocols, we can determine the possible co-utile protocols in the game tree shown in Figure 2. Since the payoff of the responder when he does not participate in the protocol (that is, when the initiator submits the query to the WSE herself) is 0, (F, D) is not a co-utile protocol because the responder's payoff is the same as in (S). The only possible co-utile protocols are:

- (F, S), which is co-utile if the responder's payoff is strictly greater than in (S), that is $u^R_{FS} > u^R_S = 0$, and (F, S) is self-enforcing, which occurs if $u^R_{FS} > u^R_{FD}$ (the responder improves her privacy by submitting the query, which happens only when the query diversifies R's query profile) and $u^I_{FS} > u^I_S$ (directly submitting the query decreases the initiator's privacy). Notice that under the previous conditions, (F, S) is also Pareto-optimal.

- (S), which is co-utile (actually just utile) when any of the following conditions is satisfied: $u^I_S > \max\{u^I_{FS}, u^I_{FD}\} = u^I_{FS}$ (by directly submitting her query, the initiator improves her privacy); or $u^I_S = u^I_{FS}$ and $u^R_{FS} = 0$ (the initiator does not decrease her privacy by directly submitting her query and the responder does not increase his privacy by submitting the initiator's query).

The following proposition summarizes the previous discussion.

Proposition 3. *Protocol (F, S) is co-utile when $u^R_{FS} > 0$ and $u^R_{FS} > u^R_{FD}$ and $u^I_{FS} > u^I_S$. In fact, under these conditions (F, S) is strictly co-utile.*

While we have been able to obtain Proposition 3 without specifying the form of the agents' utility functions, we need to do so in order to compute the agents' payoffs.

The initiator is interested in getting her query answered by the WSE and keeping her interests private w.r.t. the WSE. Thus, the initiator's utility under a protocol P can be expressed as $u_I(P) = u_I(f_I(P), p_I(P))$, where $p_I(P)$ quantifies the effect of the query submission on the initiator's privacy and $f_I(P)$ is an indicator reflecting whether protocol P gives the expected functionality to the initiator:

$$f_I(P) = \begin{cases} 0 & \text{agent } I \text{ does not get the expected functionality;} \\ 1 & \text{otherwise.} \end{cases}$$

As to the responder, his only interest is his privacy w.r.t. the WSE; therefore, his utility can be written as $u_R(P) = u_R(p_R(P))$, where $p_R(P)$ quantifies the effect of the query submission on the responder's privacy.

To measure the privacy of an agent w.r.t. the WSE, we look at the agent's query profile, Y, which is the distribution of topics of the queries submitted so far by the agent. The query profile characterizes the exposure level (that is, privacy loss) of the agent's interests towards the WSE: if an agent's profile just contains one or a few dominant topics, her/his preferences are obvious; on the other hand, if it contains a variety of topics all appearing with the same frequency, the agent's preferences remain uncertain. We measure the agent's privacy level by computing the Shannon's entropy (that is, uncertainty) of her/his profile. Thus, the effect of a query q can be measured as the variation of the profile entropy caused by q:

$$p_I(P) = H(Y_I \cup \{q\}) - H(Y_I);$$

$$p_R(P) = H(Y_R \cup \{q\}) - H(Y_R).$$

Even if the Shannon's entropy of an empty profile is not defined mathematically, by convention we set it to $\log_2 \tau$, which is the maximum value of the entropy of a distribution over τ possible topics; the reason is that an empty profile discloses no information about the interests of the agent. For non-empty profiles, the maximum entropy/privacy $\log_2 \tau$ is reached if all the topics in the profile appear with the same frequency; the rationale is that such a flat profile does not disclose any particular preferences by the agent.

4.2. Single-hop query submission game with multiple responders

So far, we have considered only two agents (initiator and responder). While this simplification is useful to illustrate how the anonymous query submission game works, it also strongly limits cooperation and therefore the privacy gain of the agents: the two agents will cooperate to submit each other's queries only if their interests are complementary. To overcome this limitation, more agents are needed. Here we assume that there is one initiator (I) and n responders ($R_1, \ldots, R_n$).

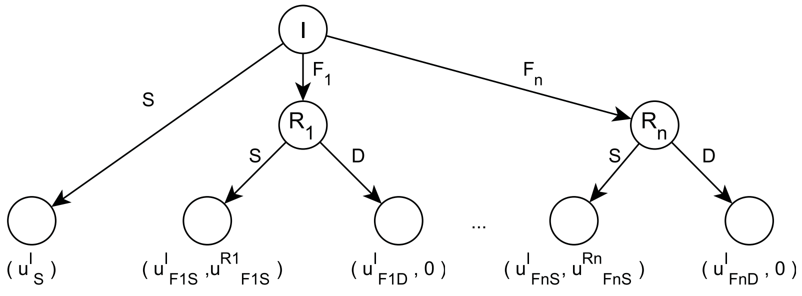

Figure 3: Tree showing the possible interactions in the single-hop anonymous query submission game with n responders. To keep the figure compact, we only represent the payoffs of the agents active in each interaction.

The interactions and utilities of the agents are the same as in the previous case; the difference is that, when forwarding the query to another agent, the initiator has to select one among the n responders. Figure 3 illustrates the possible interactions in the scenario with multiple responders. Notice that action F_i has no effect on the privacy or the functionality of responder R_j, with $j \neq i$. Thus, it is reasonable to assume that the utility of R_j, with $j \neq i$, is zero under F_i. Also, (S) has no effect over any of the responders, so that the utility of R_i under (S) should be zero for all i. Finally, when responder R_i declines the submission, the effect on the privacy and the functionality of all responders should also be zero.

If we analyze the conditions under which co-utile protocols exist in this game, we have:

Proposition 4. *Protocol* (F_i, S) *is co-utile when* $u^{R_i}_{F_iS} > 0$ *and* $u^{R_i}_{F_iS} > u^{R_i}_{F_iD}$ *and* $u^I_{F_iS} > u^I_S$.

Unlike in the game with only two agents, none of the protocols (F_i, S), for $i = 1, \cdots, n$, is strictly co-utile because each responder maximizes his utility at a different terminal node.

Since the initiator is unaware of the state of the other agents, she cannot anticipate whether a specific responder will submit or decline her query. In a game with incomplete information as this one, the initiator must check her prior knowledge on the other agents and choose the responder that she expects to be most likely to submit her query. However, following a protocol that is expected to be co-utile is not a satisfactory solution if the expectations are not fulfilled and the query is declined (in this case the protocol actually turns out to be not co-utile).

Also, it does not make sense for the initiator to stop when the query submission is declined by the selected responder. After a refusal, the initiator updates her knowledge about the other agents (by excluding the declining responder) and tries again. This trial and error process has a drawback: it can defer the

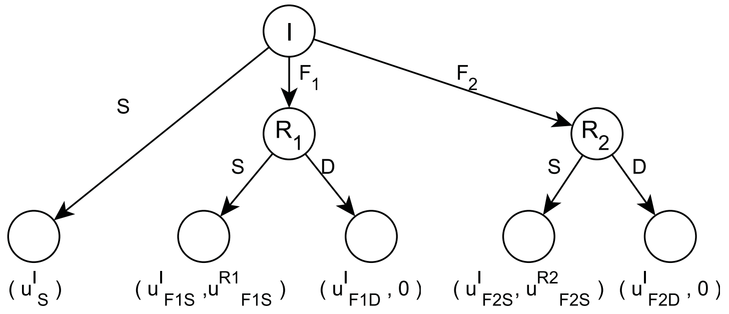

Figure 4: Tree of interactions in the case of an initiator and two responders at time $t = 0$. To keep the figure compact, we only represent the payoffs of the agents active in each interaction.

query submission too long. To prevent this, the initiator should include time among the factors that shape utility, more precisely as part of the functionality atomic utility: the initiator does not only want her query answered, she wants to get the answer within a certain time interval. To account for time, we assume that the response from the WSE is immediate when the initiator submits the query herself, while forwarding it to another agent for submission and getting the answer back implies some delay.

Each new trial implies a different game, in which the utilities are updated with respect to the previous trial game: not only the declining responder in the previous game is excluded, but the initiator's utility is decreased to reflect the passing of time. To prevent the query submission from being deferred too long, there must be a time t such that, regardless of privacy, getting the query answered at time t is always preferable to getting it answered at time $t + 1$. Thus, at time t, (S) is preferred by the initiator to (F_i, S) for any i.

Figure 4 depicts the initial game tree for a scenario with an initiator and two responders, where the initiator includes time as part of functionality. Figure 5 shows the updated game tree assuming that the initiator has played F_1 and responder R_1 has declined the submission. This is reflected in the initiator's payoffs, which are reduced by one due to the passed time: the utilities $\hat{u}^I$ in Figure 5 are smaller than the utilities u^I in Figure 4.

Let us now examine how functionality can account for passing time. Clearly, functionality can no longer be a binary function in $\{0, 1\}$. Let us assume it takes values in $[0, 1]$, so that the actual value measures the initiator's satisfaction at obtaining the query answer at time t:

$$f_I(P;t) = \begin{cases} 0 & \text{the initiator does not get the query answered at time } t; \\ \alpha_t \in [0,1] & \text{satisfaction level when getting the answer at time } t. \end{cases}$$

A stricter way to define $f_I(P;t)$ in order to prevent too long a deferral of query submission is to set a timeout, T that, if reached, makes (S) the best choice for the initiator; that is, the query answer is only useful if it comes no

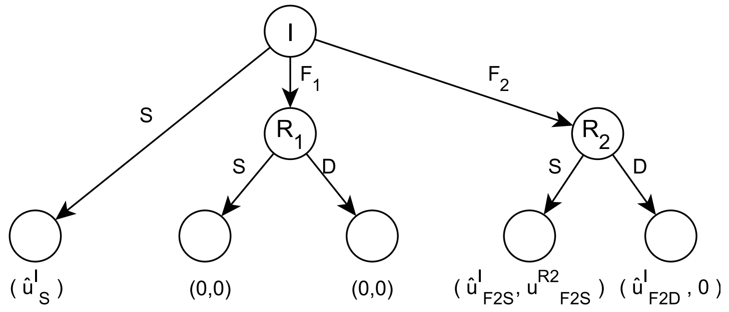

Figure 5: Updated tree of interactions in the case of an initiator and two responders at time $t = 1$, assuming that responder R_1 declined to submit the initiator's query. To keep the figure compact, we only represent the payoffs of the agents active in each interaction. Due to the elapsed time, it holds that $\hat{u}^I_S \leq u^I_S$, $\hat{u}^I_{F_2S} \leq u^I_{F_2S}$ and $\hat{u}^I_{F_2D} \leq u^I_{F_2D}$, where u^I_S, $u^I_{F_2S}$ and $u^I_{F_2D}$ are the utilities in Figure 4.

later than T. To model this behavior, we can set the functionality to $-\infty$ when time T is reached; furthermore, to abstract from the initiator's beliefs, we can consider the functionality to be constant across time until T and to become $-\infty$ thereafter:

$$f_I(P;t) = \begin{cases} 0 & \text{if } t < T \text{ and the initiator does not get the query answered;} \\ 1 & \text{if } t < T \text{ and the initiator gets the query answered;} \\ -\infty & \text{if } t > T. \end{cases} \tag{1}$$

With the above functionality, the initiator's utility does not change for times until T, so that the reasonable way to act for her is to keep trying different responders until either the query is submitted to the WSE or time T is reached. If T is reached, then the initiator submits the query herself. Although Expression (1) is reasonable for the initiator's functionality atomic utility, the actual expression depends on the initiator's preferences: if the functionality gradually decreases with time, the beliefs about the other agents' preferences become more relevant, as it may be better for the initiator to submit the query herself, even before T, if she believes the other agents are likely to decline.

4.3. Multi-hop query submission game

The single-hop query submission protocol discussed above is effective at protecting the privacy of agents against the WSE. However, since the initiator directly forwards her query to the responders, she also discloses her interests to those responders. Thus, if agents are not only interested in protecting their privacy toward the WSE, but also against their peers, the single-hop protocol is not co-utile anymore. Actually, collaborating with peer agents to avoid being profiled by the WSE can hardly be satisfactory if it has the side effect of being profiled by one's peers.

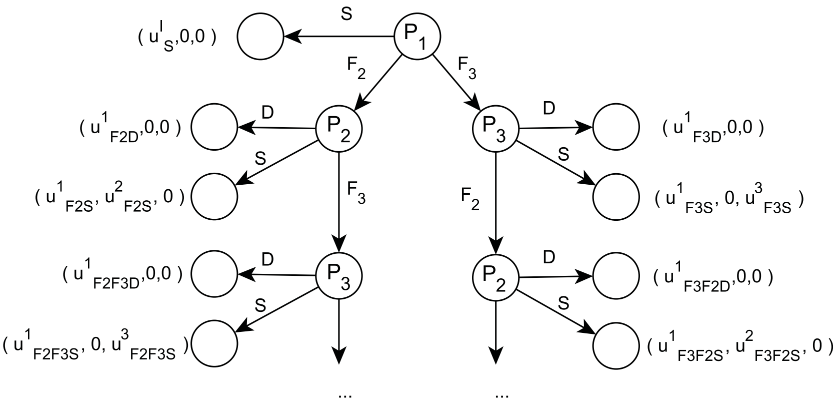

Figure 6: Tree showing the possible interactions of the multi-hop query submission game with three agents

To realize co-utility when agents want privacy against the WSE and the other agents, a more complex game needs to be designed whereby the identity of the query initiator is not only hidden to the WSE, but also to all other agents. We can achieve this by extending the actions available in the single-hop game as follows: an agent that receives a query to be submitted may now submit it, decline it or *forward it to another peer*. Specifically, forwarding the query to another agent is an alternative to declining it in case the responder does not wish to submit the query to the WSE (because doing so does not increase his privacy). The result of allowing query forwarding is that a responder agent who receives a query no longer knows whether he is receiving it from the initiator or from another responder. Figure 6 illustrates the tree of interactions of this *multi-hop* game in the case of three agents.

If in the single-hop game with multiple responders it could take too long for a query to be answered (in case many responders needed to be tried), in the multi-hop game the situation can be even worse. If agents keep forwarding the query without submitting it, the submission can be deferred indefinitely. Like in the previous game, including time in the initiator's utility is a way to deal with this potential problem. Thus, similarly as we did in the single-hop game, to abstract from the initiator's beliefs about the other agents, we assume that the initiator is indifferent to the query response time as long as the answer is received before T; when time T is reached, her utility becomes $-\infty$, which makes direct submission by the initiator the most preferable protocol.

On the other hand, in order to remain perfectly private w.r.t. other peers in the network (or any external observer), agents will also be interested to add uncertainty to the profile that results from the whole set of queries they send (those forwarded to other agents or submitted to the WSE). Note that this assumes the worst case in which a network adversary is capable of collecting all queries initiated, forwarded or submitted by each agent. We denote the profile that results from accumulating these queries by Z and, like for Y, we use Shannon's entropy to measure how disclosive it is. By incorporating this additional privacy concern into the privacy atomic utility, we have the following updated expressions that also consider the variation of the entropy that q produces in Z when q is submitted to the WSE or forwarded to another agent:

$$p'_I(P) = p_I(P) + H(Z_I \cup \{q\}) - H(Z_I);$$

$$p'_R(P) = p_R(P) + H(Z_R \cup \{q\}) - H(Z_R).$$

From a co-utility perspective, the possibility of forwarding the query to another agent masks the initiator of the query, thereby preserving her privacy versus the other agents. On the other hand, the privacy concerns of agents w.r.t. external observers make forwarding preferable to declining to any responder agent: forwarding agents also increase the privacy of their external profiles (Z) because the forwarded queries add uncertainty to the queries they really originate. Thus, in a multi-hop protocol with several forwarding hops ending in a submission, not only the initiator and the submitter take their best possible actions, but the forwarders also do. In this sense, the protocol is strictly co-utile (if we consider only the agents participating in it).

4.4. On the availability of agents with appropriate types

The above co-utile protocols are viable provided that agents with appropriate preferences/types are available. In this section, we discuss how these types are, at the end, the most common ones in a real scenario. To do so, we analyze the system in the most extreme cases.

The first extreme case happens when all agents have empty profiles towards the WSE and/or towards all the other agents because they have not (directly) submitted any query so far. We call this the "cold start" case. As discussed in Section 4.1, an empty profile offers perfect privacy (to which we assigned by convention maximum entropy in Section 4.1) because it does not disclose any preference of the agent. According to this, one may assume that agents will always prefer to rely on others to submit their queries and will decline submitting queries of others because this would decrease the entropy of their profiles towards the WSE. This "free-riding" attitude results in all requests for query submission being declined. However, this deadlock cannot be sustained very long: if the initiator has a query that does not make anyone else more private (and, thus, no one wants to forward it, let alone submit it) and if the response time is part of the initiator's utility, she will eventually be forced to submit the query herself. In this way, the initiator's profile stops being empty and its entropy falls sharply (the agent's query is completely exposed); thus, the initiator becomes highly motivated to submit queries from other agents in order to flatten her profile and thereby increase its entropy.

A similar analysis and conclusion apply to the other extreme case that we dub "happy laziness". This situation happens when most of the agents have a non-empty flat profile and, because of this, they prefer not to submit or forward queries from others. “Happy laziness” however comes to an end as soon as some agents initiate new queries: since they cannot find anyone willing to forward or submit them, they are forced to submit them themselves. As in the “cold start” case, this action unflattens their profiles and makes them willing to collaborate with others to increase again the entropy of their profiles.

5. Simulations and Empirical Results

In this section, we describe an implementation of the above described protocols and report the results (that is, agent utilities) of a set of simulations using query logs of real users.

Based on the state of the art in user profiling [32], agent profiles (Y and Z) were characterized as a vector of normalized weights, each one quantifying the agent’s interest in a certain topic. Each topic weight represents the aggregation of the semantics of the user’s queries that fall into the topic. To associate queries with topics, and to quantify their semantics, the following analysis was carried out:

- *Identify noun phrases contained in the query.* Such phrases are the semantically-richest linguistic units. For identification, we relied on several linguistic tools performing tokenization, pos-tagging and chunking [23].

- *Classify each noun phrase in a certain topic of the profile.* We looked up the noun phrase in a taxonomically structured knowledge base in order to retrieve its taxonomic ancestors. We used the Open Directory Project [22] for this purpose. For each noun phrase, we took the root of the taxonomic tree to which it belonged. For example, if looking up *MacBook*, for which ODP provides the hierarchy *MacBook* → *Portables* → *Hardware* → *Macintosh* → *Apple* → *Systems* → *Computers*, the noun phrase was classified in the *Computers* category. The set of topics defined in the user profiles matched the 15 root categories of ODP (*e.g.*, Health, Science, Business, Sports, etc.).

- *Update the weights in the user profile.* We quantified the semantic contribution of each noun phrase to the corresponding topic in the user profile by measuring the amount of information provided by the noun phrase. To do so, we measured the information content (IC) of the noun phrase as the inverse of its probability of occurrence in the Web. In this manner, the scarcer and more specific terms, which are the ones containing more information, contribute more to the topic weight in the user profile than other commoner terms with more generic semantics.

Profile weights for each topic were finally normalized according to the informativeness of the topic itself, so that the weight of the most general topics (that is, the ones covering the largest numbers of terms, and thus, being associated to the largest numbers of queries) grew more slowly than the weights of the most specific topics. See [32] for more details about the profiling process.

The simulations of the protocols were carried out with 900 agents. Each agent had a set of queries for which she wished to obtain an answer from the WSE. These queries were randomly picked from the query logs of real users available in the AOL data set [1], which consists of the actual queries performed by users of the AOL WSE during 3 months in 2006. In total, we compiled 20,928 individual queries to be initiated by the agents through the life cycle of the system. In our simulations, agents iteratively selected queries from their respective query sets and decided the actions to be taken in order to maximize their utilities.

We set up the initial profile of each agent (with regard to the WSE, Y, and for the multi-hop game, with regard to other peers, Z) to follow the distribution of topics defined by the query set associated to each agent. This makes sense, because this initial profile would be the one the agent would end up having if she directly submitted all queries in her query set and only those queries (that is, if she used no mechanism to protect the privacy of her profile).

To capture the behavior of the system more clearly, the agent types were uniformly defined. The response time was implemented with a timeout T, as described in Section 4.2. The specific value of T was chosen in a range from 5 to 40 units for all agents, where each unit corresponds to one submission request by another agent in the single-hop game and to one forwarding step in the multi-hop game.

The first reported simulation corresponds to the single-hop game. Figure 7 shows the evolution (on average over all agents) of privacy and query response time:

- The first graph shows, in the Y-axis, the average privacy of all agents w.r.t. the WSE (measured as the average entropy of the topic weights of agent profiles, $H(Y)$), as a function of the number of queries that have been submitted so far (X-axis) and the timeout T defined in the agent types (Z-axis). The horizontal plane at the top shows the upper bound for privacy (i.e., the maximum entropy of agent profiles when all the agents achieve a perfectly flat distribution of the 15 topic weights, which is $\log_2 15$).

- The second graph shows, in the Y-axis, the average response time for each query, $t(q)$, measured as the number of requests per submission, as a function of the number of queries that have been submitted so far (X-axis) and timeout T defined in the agent types (Z-axis).

We can see that the agent profiles become more and more private w.r.t. the WSE at each new iteration of the game. Since the initial agent profiles are the ones the agents would have if each of them submitted her own queries and only her own queries, this means the collaborative query submission protocol was, in most cases, co-utile (and in particular rationally followed by the agents). Due to the measurement of privacy as Shannon's entropy, privacy grows logarithmically and becomes quite flat after 10,000 queries. Even though the actual privacy is

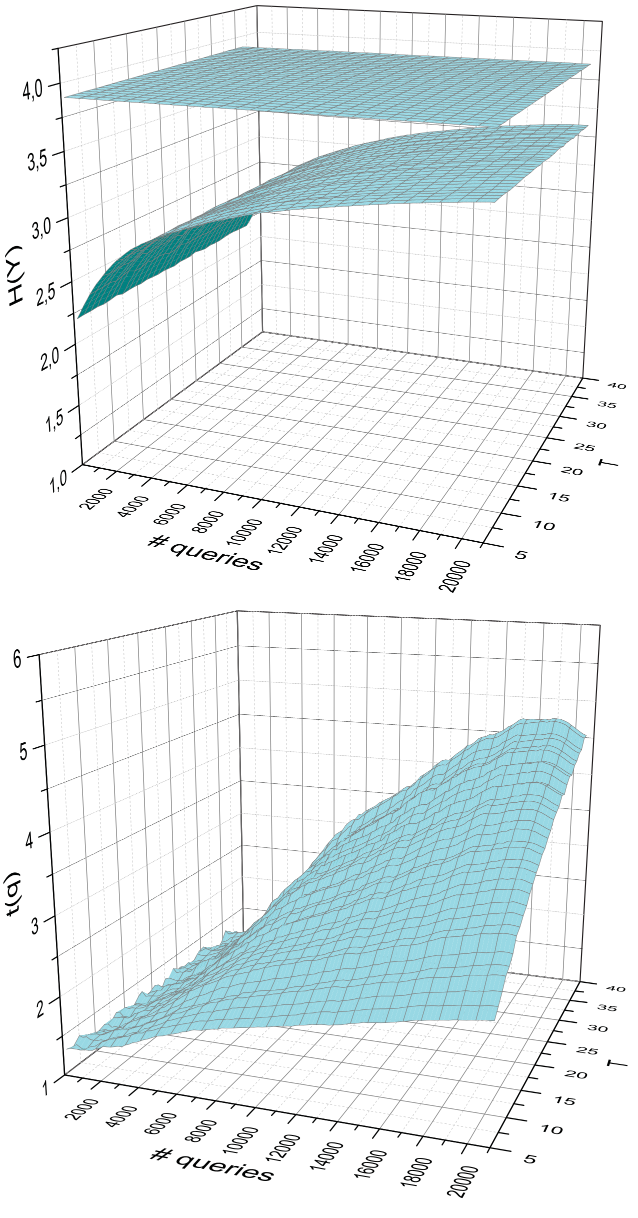

Figure 7: Evolution of agents' privacy (top) and query response time (bottom) for the single-hop anonymous query submission game

quite close to the maximum achievable value represented by the top plane (i.e., when all user profiles are perfectly flat), in practice, the limited variability of the queries (compared to the distribution of topics) makes it difficult to achieve this theoretical upper bound.

Interestingly, we can also see that the privacy increase turns out to be independent of the timeout T in the agents' types. A justification of this seemingly counterintuitive fact is that, even though a shorter timeout forces agents to submit themselves a higher fraction of the queries they originate (thereby decreasing their privacy), it also causes them to be more motivated to submit queries from other agents (thereby compensating the former privacy loss).

The response time $t(q)$ for queries also grows with the number of performed queries. The reason is that, since agents become more private w.r.t. the WSE as the number of past queries increases, they tend to decline requests from other agents more often, because many of the queries they receive do not help them increase their already high privacy. As a result, the initiator is forced to try more responders to find one who is willing to help, which increases the response time. If the initiator's timeout is long, so will be the actual response time, because the initiator will spend longer trying to find a willing responder before submitting herself her own query. On the other hand, if the timeout is short, the initiator will be forced to submit herself her own queries more often, thereby sacrificing some of her privacy; as a result, the initiator will become more motivated to submit queries from other agents (to compensate her privacy loss) and the system will become more responsive. We can conclude that a short timeout is healthy, to ensure fast response times and avoid the "happy laziness" scenario discussed in Section 4.4. Moreover, by the design of the protocol, a short timeout does not significantly impair the privacy gains of agents in the long term.

To evaluate the behavior of the system in the "cold start" scenario discussed in Section 4.4 (when none of the agents have performed any query and, thus, their profiles are still perfectly private), we performed an additional simulation in which all agents have an empty initial profile w.r.t. the WSE. Figure 8 shows the evolution of utilities in this case.

In this extreme case, the initial privacy of each agent is maximum because of her empty profile. Then, Figure 8 shows a sharp decrease because agents are forced to submit themselves their own queries in order to achieve the desired functionality. Indeed, since all agents have an initially perfect privacy, no agent is willing to submit queries from other agents because he does not gain anything by doing so. During this "cold start" stage we also observe long response times (notice that the time scale of the Y-axis of Figure 8 is much larger than in Figure 7) because of the unsuccessful attempts to get help (which involve potentially asking all available agents). Response times are also proportional to the timeout T chosen by the agents. However, as soon as agents experience a significant decrease of their privacy (because they go from a perfectly private empty profile to an exposed profile when they perform their initial query), they become highly motivated to submit queries from other agents to make up for the large privacy loss. From this point onwards, Figure 8 displays a progressive

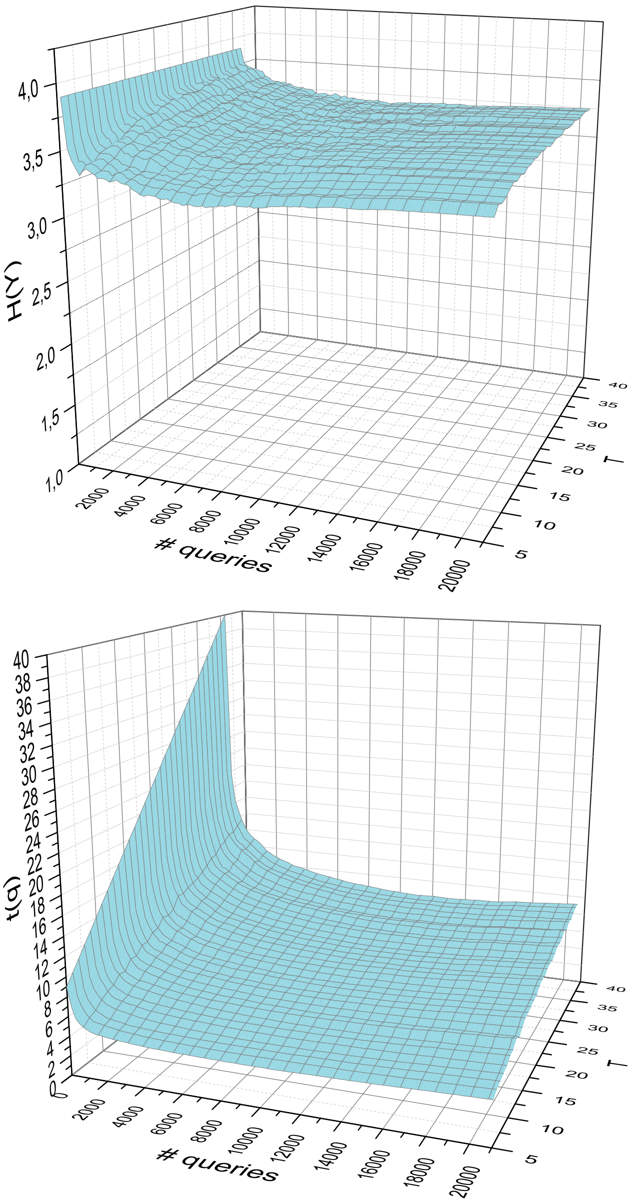

Figure 8: Evolution of agents' privacy (top) and query response time (bottom) for the single-hop anonymous query submission game with empty initial profiles

improvement of both utilities (privacy and time) that follows a pattern similar to the one in the simulation of Figure 7. The only difference is that, since now the agents start with empty profiles, reaching a high privacy level is easier for them than in the previous case (in which they started with an initial profile corresponding to the distribution of the queries they were actually interested in). In our simulation, the turning point between the "cold start" stage and the normal system operation stage (with high privacy and reduced response times) occurs when around 9,000 queries have been performed (that is, 10 queries per agent, on average).

Finally, we ran a simulation with the same set of agents, types and query logs for the multi-hop game. As discussed in Section 4.3, the possibility to forward queries masks the initiator's identity and protects her privacy against the other agents. Moreover, in this case, an additional privacy goal has been considered and quantified: the agent's privacy w.r.t. external observers, which is measured as the entropy of the profile Z formed by the set of queries sent out by each agent to the WSE or to other agents. The initial profile Z has been taken to be equal to the initial profile Y of the agent w.r.t. the WSE, because we assume that the queries sent out by the agent prior to joining the game are those directly submitted to the WSE.

Figure 9 shows the evolution of the partial utilities for the multi-hop game.

The analysis of the privacy w.r.t. the WSE and query response times we made for the single-hop game with non-empty initial profiles is also applicable here. Specifically, the improvement the agents' privacy with the number of performed queries shows that the (co-utile) collaborative query submission protocol is followed in most cases. The only difference is that now the timeout T is even more important than in the single-hop game because, due to the query forwarding step, agents may need to wait longer until they realize that they have failed to find help to submit a query. This can be seen by comparing the maximum response time for the longest timeouts of both games (Figure 7 vs Figure 9), which is higher in the multi-hop game. The privacy of the agents' profile w.r.t. external observers ($H(Z)$) behaves similarly to the privacy of their profile w.r.t. the WSE ($H(Y)$). However, $H(Z)$ tends to grow faster because it increases not only when queries are submitted to the WSE, but also when they are forwarded to other peers; this makes forwarding queries preferable to declining them in most cases, and thus, makes collaboration self-enforcing. Indeed, since now a query submission may involve several peers, an individual query can produce privacy gains (in the Z profile) to several agents (forwarders and submitter) and, thus, it provides a greater total gain.

6. Conclusions and Future Work

We have studied co-utility, which is a new paradigm to design self-enforcing protocols that make mutual help between rational agents the best strategy. Hence, co-utility has the potential to boost social welfare in a great number of interactions, both in information society and in the society at large.

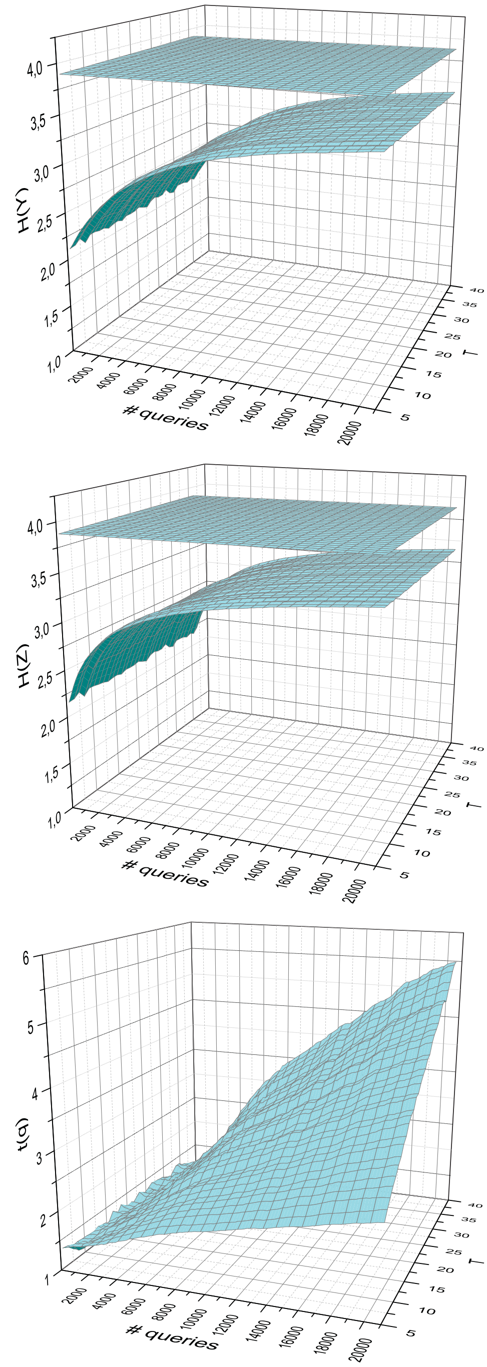

Figure 9: Evolution of utility outcomes for the multi-hop anonymous query submission game: agents' privacy w.r.t. the WSE (top), w.r.t. external observers (middle) and query response time (bottom)

To make co-utile collaborative protocols viable, the protocol designer should configure the game and define agent actions by carefully considering the agents' interests/utilities, as we have illustrated in the query submission scenario. In this respect, in real-life scenarios, the actions of agents may have an associated cost (for example, submitting or forwarding a query costs bandwidth and/or money), which may render collaboration not rational anymore. To neutralize such negative payoffs/costs, artificial rewards or penalties can be employed by relying on a reputation mechanism. For example, the reputation of an agent may increase when she incurs a cost to help others and decrease when she causes a cost to the ones who help her. Then an agent would only receive help from others if she had accumulated at least a certain level of reputation by helping others. To compute and manage agent reputations, one may rely on distributed anonymous reputations mechanisms (which are decentralized and secure against reputation tampering) [33, 15].

As future work, an interesting theoretical goal is to extend our framework to sequential games with imperfect information, as mentioned in Section 2. On the practical side, we will implement and test different distributed reputation mechanisms to find the best option in terms of computation, fairness, collusion security and frameproofness [12]. Moreover, we also plan to study co-utile protocol design in a variety of application domains (in principle, any peer-to-peer game lends itself to co-utile protocol design). Promising applications are:

- Distributed computing, including cloud provider federations, the Internet of Things, etc. In distributed scenarios, a number of autonomous or entirely independent systems are supposed to interact in certain prescribed ways. Since those systems often operate in different legal jurisdictions, compelling them to perform as agreed (even if stated in a contract) may be difficult. Making prescribed interactions rationally sustainable would be hence interesting. Co-utile protocols induced by artificial incentives such as reputation or quality of service could go a long way in this direction.

- The collaborative economy, including crowdsourcing [21], ridesharing [29], etc. The lack of a common legal framework and, hence, the lack of trust among peers hamper a more generalized take-off of the collaborative economy. Especially interesting would be a fully distributed collaborative economy, to empower the individual agents and overcome the current oxymoron of some platform-owning companies acting as oligopolies in the collaborative market. Just as in the case of distributed computing, suitable artificial incentives like distributed reputation could be leveraged to advance towards a distributed collaborative economy.

Acknowledgments and Disclaimer

Thanks go to Oana Ciobotaru for discussions in the early stages of this work. Funding by the Templeton World Charity Foundation (grant TWCF0095/AB60

"CO-UTILITY") is gratefully acknowledged. Also, partial support to this work has been received from the Government of Catalonia (ICREA Acadèmia Prize to J. Domingo-Ferrer and grant 2014 SGR 537), the Spanish Government (projects TIN2014-57364-C2-1-R "SmartGlacis", TIN2015-70054-REDC and TIN2016-80250-R "Sec-MCloud") and the European Commission (projects H2020-644024 "CLARUS" and H2020-700540 "CANVAS"). The authors are with the UNESCO Chair in Data Privacy, but the views in this paper are the authors' own and are not necessarily shared by UNESCO or any of the funding bodies.

References

[1] AOL Search Data Mirrors. Available at `http://gregsadetsky.com/aol-data/` (last accessed: Jan. 17, 2016)

[2] R. R. W. Brooks, C. M. Landeo and K. E. Spier. Trigger happy or gun shy? Dissolving common-value partnerships with Texas shootouts. The RAND Journal of Economics 41(4)(2010) 649-673.

[3] F. Buccafurri, G. Caminiti and D. Rosaci. An ASP-based approach to dealing with agent perception failure. AI Communications (AI COMM) 21(1)(2008) 4969.

[4] F. Buccafurri, G. Caminiti and D. Rosaci. Logic programs with multiple chances. In: Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006), pp. 347-351.

[5] C. Buragohain, D. Agrawal and S. Suri. A game-theoretic framework for incentives in P2P systems. In: Proceedings of Peer-to-Peer 2003, pp. 48-56.

[6] J. Castellà-Roca, A. Viejo and J. Herrera-Joancomartí. Preserving user's privacy in web search engines. Computer Communications 32(13-14)(2009) 1541-1551.

[7] G. Chalkiadakis, E. Elkind and M. Wooldridge. Cooperative game theory: basic concepts and computational challenges. Intelligent Systems 27(3)(2012) 86-90.

[8] Y. Dodis and T. Rabin. Cryptography and game theory. In: N. Nisan et al. (eds.), Algorithmic Game Theory. Cambridge University Press, 2007, pp. 181-205.

[9] J. Domingo-Ferrer, A. Solanas and J. Castellà-Roca. $h(k)$-Private information retrieval from privacy-uncooperative queryable databases. Online Information Review 33(4)(2009) 720-744.

[10] J. Domingo-Ferrer and Ú. González-Nicolás. Rational behavior in peer-to-peer profile obfuscation for anonymous keyword search. Information Sciences 185(2012) 191-204.

[11] J. Domingo-Ferrer and Ú. González-Nicolás. Rational behavior in peer-to-peer profile obfuscation for anonymous keyword search: the multi-hop scenario. Information Sciences 200(2012) 123-134.

[12] J. Domingo-Ferrer, O. Farràs, S. Martínez, D. Sánchez and J. Soria-Comas. Self-enforcing protocols via co-utile reputation management. Information Sciences 367(C)(2016) 159-175.

[13] E. J. Friedman, J. Y. Halpern and I. Kash. Efficiency and Nash equilibria in a scrip system for P2P networks. In: Proceedings of the 7th ACM conference on Electronic commerce. ACM, 2006, pp. 140-149.

[14] D. C. Howe and H. Nissenbaum. TrackMeNot: resisting surveillance in web search. In: Lessons from the Identity Trail: Privacy, Anonymity and Identity in a Networked Society. Oxford University Press, 2009, pp. 417-436.

[15] S. D. Kamvar, M. T. Schlosser and H. Garcia-Molina. The Eigentrust algorithm for reputation management in P2P networks. In: Proceedings of the 12th International Conference on World Wide Web. ACM, pp. 640-651, 2003.

[16] H. W. Kuhn. Extensive games and the problem of information. In: Contributions to the Theory of Games, II, pp. 193-216, 1953. Reprinted in: H.W. Kuhn (ed.), Classics in Game Theory. Princeton University Press, 1997.

[17] K. Leyton-Brown and Y. Shoham. Essentials of Game Theory: A Concise, Multidisciplinary Introduction. Morgan & Claypool, 2008.

[18] R.D. Luce and H. Raiffa. Games and Decisions: An Introduction and Critical Survey. John Wiley & Sons, 1957.

[19] E. Maskin. Nash equilibrium and welfare optimality. The Review of Economic Studies 66(1)(1999) 23-28.

[20] M. Murugesan and C. Clifton. Providing privacy through plausible deniable search. In: Proceedings of the SIAM International Conference on Data Mining (SDM 2009). SIAM, pp. 768-779, 2009.

[21] A.N. Turi, J. Domingo-Ferrer, D. Sánchez and D. Osmani. A co-utility approach to the mesh economy: the crowd-based business model. Review of Managerial Science (to appear).

[22] Open Directory Project. Available at: `http://www.dmoz.org` (last accessed: Jan. 17, 2016)

[23] OpenNLP Maxent Package. Available at: `http://maxent.sourceforge.net/about.html` (last accessed: Jan. 17, 2016)

[24] M. Osborne and A. Rubinstein. A Course in Game Theory. MIT Press, 1994.

[25] M. S. Papamarcos and J. H. Patel. A low-overhead coherence solution for multiprocessors with private cache memories. In: Proceedings of the 11th Annual International Symposium on Computer Architecture-ISCA'84. ACM, 1984, pp. 348-354.

[26] N. Nisan. Algorithmic mechanism design. In: Handbook of Game Theory. Elsevier, 2014, pp. 477-516.

[27] R. Rahman, T. Vinkó, D. Hales, J. Pouwelse and H. Sips. Design space analysis for modeling incentives in distributed systems. In: Proceedings of the ACM SIGCOMM 2011 Conference. ACM, 2011, pp. 182-193.

[28] M. K. Reiter and A. D. Rubin. Crowds: anonymity for web transactions. ACM Transactions on Information and System Security-TISSEC, 1(1)(1998) 66-92.

[29] D. Sánchez, S. Martínez and J. Domingo-Ferrer. Co-utile P2P Ridesharing via Decentralization and Reputation Management. Transportation Research Part C: Emerging Technologies 73 147-166.

[30] B. Schneier. The value of self-enforcing protocols. Threat-Post, Aug. 10, 2009. Available at `https://threatpost.com/value-self-enforcing-protocols-081009/72980/` (last accessed: Jan. 17, 2016)

[31] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. The Journal of Finance 16(1)(1961) 8-37.

[32] A. Viejo, D. Sánchez and J. Castellà-Roca. Preventing automatic user profiling in Web 2.0 applications. Knowledge-based Systems 36(2012) 191-205.

[33] B. Yu and M. P. Singh. An evidential model of distributed reputation management. In: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems. ACM, 2002, pp. 294-301.