Collaborative Scheduling: Threats and Promises

Rachel Greenstadt and Michael D. Smith
Harvard University
{greenie,smith}@eecs.harvard.edu

June 5, 2006

Abstract

For people to entrust their private information to modern collaborative tools, they will want some assurance of the extent to which this information will be protected. Accurately measuring the amount of privacy lost by different collaborative technologies is a critical step in providing such assurances. Unfortunately, the identification of intuitive and useful metrics for privacy loss has proven difficult. In this paper, we analyze the problems with the existing metrics for distributed constraint optimization, and based on our findings propose a new metric for measuring privacy loss that avoids these problems. In particular, our metric quantifies the amount of private information definitively lost when given an economic adversarial model. We show the importance of a defined threat model in measuring privacy loss and the dangers of metrics based on too much uncertainty. We develop and demonstrate our metric for the specific problem of collaborative meeting scheduling and use it to show the effects on privacy of network topology in distributed constraint optimization.

1 Introduction

Personal assistant agents are a tool for collaboration in businesses, office environments and research organizations [1–3]. We see evidence that their use is entering the mainstream in Microsoft's acquisition of Groove Networks collaboration software, in Google's calendaring application CL2, closely integrated with Gmail, and in Apple's support of iCal. The DARPA Coordinators project is evidence of the importance to the military of automated scheduling and rescheduling of coordinating agents [4].

The promise of these technologies is that they will enable us to schedule meetings and coordinate resources in optimal and automated ways. To fulfill this promise, the agents in these technologies must be given potentially private information about the users the agents represent. This private information might be a person's salary, a firm's capabilities, or an employee's scheduling preferences. In the course of negotiation and conflict resolution, the agents must exchange some amount of the users' private information in computing a good group outcome. The best technologies would reveal as little of the users' private information as possible while optimizing for the global social welfare.

Maintaining users' privacy has been a fundamental concern of work in distributed constraint optimization (DCOP) [3, 5, 6]. Several recent approaches to DCOP [3, 5] attempt to enable distributed conflict resolution and coordination while maintaining users' privacy. Yet, achieving less privacy loss has turned out to be more difficult than first envisioned. In particular, recent privacy loss analysis [7, 8] has shown that some proposed DCOP algorithms may actually preserve less privacy than a centralized approach. In the centralized approach, the single agent computing the global solution learns everything about everyone.

This counter-intuitive result has pushed the field to be more principled in its approach to minimizing privacy loss in DCOP. In particular, the field has begun to develop quantitative metrics for analyzing the privacy loss in existing DCOP algorithms [7,9–11]. If we can measure and understand the cause of privacy loss in existing DCOP algorithms, then we might be able to envision new techniques to reduce these losses or more accurately define appropriate use cases for the existing techniques.

Unfortunately, the metrics currently used exhibit two important problems. They don't capture intuitive notions of privacy loss well, and they can yield conflicting conclusions when comparing the privacy loss of two DCOP algorithms.

In this paper, we show how the current metrics have fallen short, and based on what we have learned propose a new metric for measuring privacy loss in DCOP algorithms. Our work builds on the *Valuation of Possible States* (VPS) framework [7], which was designed as a quantitative model for comparing privacy loss metrics and developing new metrics. Using this framework, we show that it is possible to define a metric that quantitatively tells us qualitative information about the privacy loss inherent in a DCOP algorithm and the likelihood of additional losses resulting from further adversarial action. By splitting apart the definitive (i.e. an aspect of the algorithm used) and the probabilistic (i.e. an aspect of the adversary we assume) into two separate components of a metric for privacy loss, we are able to avoid the problems found with previous metrics.

For clarity of discussion, we focus our examples on the problem of meeting scheduling and the users' desire to keep their meeting and time slot preferences private. Our ideas, however, are more broadly applicable. For example, our techniques should be useful in resource allocation problems, where one might imagine several government, corporate and nonprofit groups coming together to help in a disaster relief effort. They all wish to contribute resources that will lead to the optimal end, but they are mutually distrustful in most of their other endeavors and do not want the other entities to know the details of their individual holdings and constraints.

Though finding the right metric for privacy loss in DCOP may be less exciting as proposing a new DCOP algorithm, understanding how such algorithms lose private information in multiagent negotiations now, before the applications are widespread and mainstream, is crucial. People make bad privacy decisions in part because they discount the future danger of abuse against current convenience [12]. If people make decisions when both convenience and abuse are in the future, then these considerations may be weighed more rationally. On the other hand, if a system with high privacy loss becomes widely deployed and adopted, the argument for an accurate metric for privacy loss will be harder to make. In particular, the network effects that help increase the dominance of the entrenched model will make it difficult for privacy-sensitive in-

dividuals to opt out, and in some cases, it may be impossible for those individuals to opt out when their superiors in an organization (who may not particularly value their employees' privacy) dictate usage of the entrenched model. Privacy will consequently become less of a design goal and accurate privacy loss metrics less important.

Fortunately, adoption is not yet that widespread and private solutions may be adopted if they can be measured against nonprivate ones. In Section 2, we present a critique of existing privacy metrics. Then, in Section 3, we define a new metric that addresses the weaknesses of existing metrics. In Section 4, we demonstrate the use of our metric in analyzing DPOP and Adopt, two popular DCOP algorithms. Lastly, we analyze related work and conclude with future directions.

2 Analysis of Existing Metrics

Privacy has been a motivating factor in DCOP from the beginning of the field [3, 13], when researchers assumed that a distributed solution would automatically yield more privacy than a centralized one. Work on constraint satisfaction has also been concerned with measures of privacy loss [9, 10]. The most comprehensive work to date, by Maheswaran and his colleagues at USC, introduced the Valuations of Possible States (VPS) framework [7, 8].

This work models the private information of an individual agent G as a state $g \in S$, where S is a set of possible states that an agent may occupy. VPS defines privacy loss as the difference between an adversary's view of the agents' states before negotiation (i.e., before a DCOP algorithm is run) and the resulting view after negotiation. VPS is able to model existing metrics by leaving open the question of how to value a specific change in viewpoint about an individual agent G and how to aggregate the total privacy loss of all agents involved in the negotiation.

As a concrete example, consider the application of VPS to the meeting scheduling problem. To begin, we assume that an adversary A learns from observing the negotiation that G exists in one of $R_G \subset S$ remaining states. The adversary A may be able to eliminate states from his initial beliefs about G to achieve R_G by capturing some subset of the messages sent during the negotiation or through inference on captured messages and the final solution. Given such an application, Maheswaran et al. [7] shows how it is possible to compare the quantitative results produced by six previously described metrics of privacy loss.

In particular, Maheswaran et al. [7] classified the investigated metrics by the manner in which they partition the state space (i.e., a single large state space for all timeslots, or a unique state space for each timeslot) and by the degree of nonlinearity applied to the state space being evaluated (i.e., linear, asymptotic, or logarithmic). If we define the total number of states $|S|$ as s, the number of states remaining at the end of the inference $|R_G|$ as r, and scale privacy loss to range between 0 and 1, the six metrics are as follows: Starting with the single large state space, *LinearS* directly measures the number of states not eliminated: $\frac{s-r}{s-1}$. *GuessS* gives the probability that the adversary A will be able to guess the state g accurately from the among the eliminated states: $\frac{1}{s-1} * \frac{s}{r}$. *EntropyS*, which was introduced by Franzin et al. [9], considers privacy loss from an information-theoretic perspective: $1 - \frac{lg(r)}{lg(s)}$. The other three metrics (*Lin-*

earTS, *GuessTS*, and *EntropyTS*) are defined analogously, but treate each timeslot in the meeting schedule separately. To create a single value r for the formulae previously described, the authors simply took the arithmetic average of the number of eliminated states over all timeslots.

A significant contribution of the VPS framework was that it allowed researchers to investigate whether distributed solutions did actually produce less privacy loss than centralized solutions, using whichever metric they chose. The original VPS work and some subsequent work [8, 14] demonstrated that some distributed algorithms lost more privacy than a centralized algorithm; all agents learning some private information cannot be said to be more private than one centralized agent learning all the private information. More interesting from the point of this paper was the observation that

> "[O]ne needs to carefully examine the metrics chosen to measure privacy loss; the qualitative properties of privacy loss and hence the conclusions that can be drawn about an algorithm can vary widely based on the metric chosen." [8]

The VPS framework clearly points out that we not only have multiple proposed metrics for privacy loss, but the metrics we have do not quantitatively tell us qualitative information about the likelihood of disclosure of the private data during negotiation. This makes it very hard for us to use these metrics to identify the best performing (in terms of privacy loss) algorithms, to test the effects on privacy loss of changes to existing algorithms, and to understand the qualitative tradeoff between convenience and privacy.

We postulate that these problems are caused by three main issues associated with each of the existing metrics. The rest of this section explains each of these issues in turn. By understanding and raising the awareness of these issues, we can define new metrics that are intuitive and aid in the development of technologies for DCOP that produce optimal or near-optimal solutions without releasing more private information than necessary. Section 3 proposes one such metric.

2.1 Unproductive Aggregation

One surefire way to produce a number that defies intuition and leads to counter-intuitive results is to aggregate values because you can and not because you should. This problem appears in several different ways in the metrics described earlier.

Let's begin by considering the case of aggregating probabilistic information with partial information. In the whole state space model (LinearS, GuessS, and EntropyS), certain knowledge (e.g., agent G's valuation for timeslot 0 is 3) is aggregated with the overall uncertainty of the complete state vector. As an illustrative example, assume that you would like to evaluate the privacy loss in two algorithms a_1 and a_2 to determine which one better preserves an agent G's privacy. We start each negotiation with the belief that agent G is in one of 216 states for a scenario with three timeslots and six valuations per timeslot (i.e., every state is equally likely). At the end of the algorithm a_1, an adversary has learned that agent G is in one of 10 possible states, while after observing algorithm a_2 the adversary learns that agent G is in one of 8 possible states.

Since algorithm a_2 revealed 208 states while a_1 revealed only 206 states, the metric implies that a_1 outperforms a_2. However, if the first element of all 10 state vectors is 3 (i.e., it is known that agent G's valuation for timeslot 0 is 3) in R_G set for a_1, but no such definitive information for a particular timeslot can be extracted from the remaining 8 states in R_G for a_2, it is not at all clear that agent G experienced less privacy loss under a_1 than a_2.

This is the situation that the per-timeslot metrics are meant to address; however, the per-timeslot metrics average the privacy loss values across all timeslots to produce the privacy loss value for an individual agent. As such, these metrics produce a similar conflating of probabilitic and certain information.

This problem is only exacerbated when the metric for the privacy loss of N agents in a DCOP is computed by an aggregation of the privacy loss between each pair of agents. For example, when the method of aggregation is arithmetic mean, as in one of the earlier metrics, we come upon the counter-intuitive result that an algorithm which reduces the state spaces of all agents by 50% is equivalent in terms of privacy loss to another algorithm which reveals one half of the agents' valuations completely and nothing about the other half.

2.2 Unclear Mapping to Actual Privacy Threats

Aggregation from pairwise measures of privacy loss between individual agents to produce the privacy loss for the entire negotiation uncovers another problem with the previously described metrics: It is not at all clear how probabilistic statements map to actual privacy threats, even assuming that they are not obscured by unproductive aggregation. The various metrics give some viewpoint on what it means to reduce an agent's valuation vector from, say, 216 possibilities to 10, but it hard to understand if this difference has meaning qualitatively in terms of actual threats. The metrics as currently described provide no help in answering this question.

Worse yet, computing privacy loss by aggregation of pairwise losses can sometimes point us in an obviously wrong direction when we lose sight of the threats to the entire system. In particular, a plausible argument based on pairwise aggregration for the privacy loss due to centralized constraint optimization is $1/N$ for a negotiation involving N agents. Taken at face value, this result implies that a centralized algorithm approaches zero privacy loss as N increases. When in actual fact, as N grows, more agents are sending their private information across the entire network to the one centralized node who will calculate the result. If an adversary is able to capture the messages sent to the central node, the private information of more people will be exposed as N increases.

2.3 Difficulty of Reasoning Under Uncertainty

It is a well-known fact that people are more likely to make bad decisions when asked to reason about large amounts of uncertainty [15]. Even without the troubles of aggregation, there is something deeply unintuitive about probabilistic metrics that measure privacy loss as the reduction of uncertainty about a state, without achieving certainty. No comprehensible metric can capture everything about the privacy loss of an algorithm. While there is undoubtably some information leakage when the adversary knows that

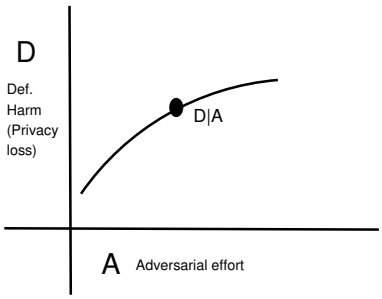

Figure 1: Visualizing privacy properties on two-dimensional plane

agent G's valuation is not x, it is less crucial than when the adversary determines that G's valuation is y. If we conflate dangers which are immediately relevant once information is revealed with dangers which might exist should some new information become available, we force people to reason about these systems under large amounts of uncertainty, where they are likely to make bad decisions.

3 A New Metric: $D|A$

We argue that it is useful to consider as privacy loss only information that, when released, can lead directly to harm. We omit from our metric probabilistic information that can only lead to harm if combined with further information sources. This gives us a clear picture of our threat and does not force users to reason about threats under uncertainty.

Our metric accomodates the fact that the amount of definitive harm that can be done with access to the messages from a given DCOP algorithm can vary widely depending on the resources, priorities and behavior of the adversary. The adversary may choose to spend computational resources on running expensive inference algorithms on the messages, spend time seeking out extraneous information to combine with that from the algorithm in external channels [16], or risk exposure by attempting to collude with other agents or actively attacking the protocol. These types of threats can be characterized economically, as explored by Schechter [17]. In summary, our metric calculates something definitive about the harmful privacy violations that occur for a specific system with an adversary spending a given amount, thus every evaluation of an algorithm must come with a defined threat model. Our metric places the definitive harm D that occurs as a result of privacy loss in the context of some A, the economic model of our adversary. We can phrase the privacy properties of an algorithm as $D|A$ and plot points (A, D) on a two-dimensional plane to visualize these properties, as shown in Figure 1. The shape of the curve may also tell us about interesting privacy loss properties of the algorithm.

In order to develop concrete metrics we must characterize the information we care about protecting, then characterize the expeditures that the adversary will have to outlay in order to obtain it. In the following section, we discuss how to do this in the collabo-

rative scheduling domain, beginning with identifying private informaiton. Clearly, the information that is private will be different in other problem domains, but the general approach still applies.

3.1 *D* for Definitive Harm

In order to measure *D* we must identify the private information that can cause harm. We have to ask: what is definitely known, what is private, what is worth keeping private?

In the case of collaborative scheduling there are a number of things that people might be concerned about. We may want to hide:

1. Our valuations of the utility of certain meetings, especially as compared to other meetings (e.g., Bob may not want Alice to know that he finds his meeting with Carol more important than the one with her).

2. Our valuations of various time slots, as it might reveal personal events outside of the meetings to be collaboratively scheduled (e.g., a medical appointment or support group meeting).

What we mean by definitive measures is that reducing Alice's possible valuations for noon on Thursday (Timeslot 1) from 6 possibilities to 3 does not count, but reducing it to one possibility (learning that it is 4, for instance) does. When the adversary learns some private information with certainty, we take notice.

In order to measure the privacy of various DCOP algorithms we must understand how these types of pieces of private information can be exposed while running a DCOP algorithm. The following process is representative of how information is communicated in many DCOP algorithms:

- Problem setup: The domain of the variables (i.e., timeslots during which meetings may be scheduled) and the domain of the valuation function for constraints (i.e., ways in which people can express their preferences for certain assignments of meetings to timeslots). These are public. If they are not well understood by all parties concerned, a meaningful solution cannot be found.

- Problem setup: The constraint graph. In the meeting scheduling domain, this is the graph where vertices are agents and edges represent meetings that both agents attend. We assume agents must know about edges that they share with others, whether they know about other edges (or how many other edges) is a property of the algorithm used.

- Message communication: The smallest message transferred contains a single tuple of a *context* and an associated *valuation*. The context is a set of possible assignments of variables to values and the valuation is the utility of that context to an agent or group of agents. In the scheduling domain, this is an assigment of meetings to timeslots. These messages may reveal personal information.

- The solution: The final times and participants of meetings that an agent schedules, or, in more general terms, the assignment of values to the variables. A solution may reveal personal information.

To understand if personal information is definitively lost in message communication, we need to determine if the messages reveal actual information that can cause harm. In this case, we deem harmful information to be information about the valuation of an exact time slot or the valuation of meeting where such a meeting is valued. Any given row of an assignment/valuation table may not reveal this information, though the whole table of all possible assignments should reveal the personal information for every timeslot and every meeting.

In order to translate these ideas into a number for D, we take each piece of private information (each private valuation of a timeslot or meeting) and assign it a fraction of the whole. If the information is revealed to the adversary A then that is counted as privacy lost. At the end, we will have a number between 0 and 1 representing privacy loss in the algorithm.

3.1.1 VPS formalism for the D metric

In VPS, each agent's private information is represented as a state $s_i \in S_i$, where S_i is a set consisting of all possible values for s_i. In order to express our metric in VPS, we consider s_i to be a tuple of values $t_1 \cdots t_n$ where each $t_x \in s_i$ is a discrete piece of private information (t_x might represent Alice's valuation for timeslot x). In the meeting scheduling example, t_x might be agent i's valuation for a particular timeslot. We express the set for possible values for a piece of personal information t_x as $S(t_x)$, a subset of S_i.

To express our metric in VPS, we begin with Equation (1) which calculates the privacy loss of each agent i, $\mathbb{V}_i(\mathbb{P}_i(S_i))$. Equation (1) simply sums up the pieces of i's private information that are definitely known (with probably 1) by at least one other agent j.

$$\sum_{x=1}^{|s_i|} I_{\sum_{j \neq i} \sum_{t \in S(t_x)} I_{\mathbb{P}_i^j(t)=1}} \tag{1}$$

The outermost summation in Equation (1) checks each piece of i's private information t_x in turn, adding 1 to the total when another agent j definitely knows the value of t_x. The indicator function I ensures that we count each piece of lost information at most once, even when more than one other agent j knows it. The innermost summation checks each possible value t of a piece of private information t_x to determine whether an agent j believes with probability 1 that agent i's value for t_x is t. Here, we assume that an agent j never misrepresents its knowledge or makes a mistake in its inference calculation.

Equation (2) aggregates the privacy loss of each agent i to calculate the system-wide privacy loss by summing up the privacy losses of the individual agents and then dividing by the amount of private information in the system.

$$\frac{\sum_{\forall i} \mathbb{V}_i(\mathbb{P}_i(S_i))}{\sum_{\forall i} |s_i|} \tag{2}$$

This metric avoids many of the problems of previous metrics. Since we do not measure the reduction of uncertainty, but instead focus on the achievement of certainty, our metric is clear and easy to reason about. Much of our critique of previous metrics deals with the ambiguity resulting from aggregating pairwise measures of privacy loss between agents into a single number. We instead consider each piece of private information discretely and consider it lost if the adversary A can know it. Previous metrics had many factors pulling the ultimate number in many directions such that it became hard to interpret. Our D metric encompasses less information, but achieves more clarity.

3.2 A for Adversary

In order to be meaningful, our value D of definitive harm must be placed in the context of an adversary A. We have to consider actions that an adversary may take so that we can translate this effort into a linear notion of economic expenditure. There are a number of actions an adversary can take if they are interested in determining private information in a distributed constraint optimization environment. Each of these actions will have some cost that can be represented in a model of A.

- The adversary may view all the messages which are sent to a single agent in the course of an algorithm, to see if they reveal anything interesting. We consider this to be the adversary that expends a negligible cost in obtaining information.
- More information may be available from these messages than can be trivially observed from the messages received. Our adversary might download a program, say, "StalkerPro" and run sophisticated inference techniques on the messages received to extract additional information from them.
- Furthermore, our adversary might have outside knowledge about the constraints of other agents (e.g., "Dan and Sally are meeting with Alice at 4pm, I heard them talking about it in the hall"), with which he could tune "StalkerPro" to provide more insight.
- Our adversary could collude with other agents to determine information (e.g., several agents might gather their messages together and see which one of their coworkers is interviewing elsewhere).
- Our adversary could actively manipulate the message stream[1]

[1] For instance, the adversary might ask for valuations that he doesn't actually need to calculate the optimal solution

Correlating these actions to economic expense is a considerable challenge, worth undertaking, but beyond the scope of this work. In the examples in Section 4, we assume the adversary expends negligible effort, simply waiting for harmful information about other agents to be handed to him by the algorithm. If we can force the adversary to expend resources to violate privacy, then we will have gained something. As we will see, analyzing algorithms using this measure alone gives us valuable insights about the effect of topology on privacy in DCOP algorithms.

We realize that our emphasis on definitive privacy loss is counter to prevailing notions of security in computer science. For instance, in cryptography, *semantic security* poses that all information beyond a specific threshold is considered a risk. These metrics accounting for partial information are useful, it is claimed, when the power of the adversary is uncertain. In truth, they are only useful when the system designers are able to claim that no information is released. There are designs for algorithms that are secure in this way [18, 19], however they make use of secure function evaluation techniques which are highly inefficient. In order to obtain a somewhat private solution which does not make use of these techniques, it is expected that some information will be leaked. If we can characterize the nature and capabilities of the adversary, we can bound the amount of harm he can do. If, on the other hand, we have mischaracterized our adversary, it is likely that a bound on the partial information he has received will be meaningless in this new category. For example, if we know that, under our model, the adversary was able to reduce some state space by 30%, then if the adversary has been mischaracterized, meaning he colludes with more agents than expected or he possesses side channel information that he brings to bear on the problem then he *may* be able to determine actual private information. We argue that his likelihood of doing so may not be that much improved if he reduced the state space by 70% under previous measures.

Attempting to characterize the adversary and the harm he can do in absolute terms, rather than using economic methods, makes it hard to tell with partial information what an adversary needs to do to turn it into real information. By contextualizing our metric—$D|A$—we can make more meaningful statements.

4 Example in Collaborative Scheduling

In this section, we demonstrate the utility of our metric in a simple collaborative scheduling scenario. We first describe the problem: describing the constraint graph, private valuations, etc. Then we explain how to apply our metric to the DPOP algorithm run over this scenario. We use our metric to measure the effect on privacy loss of changing the topology of the agents. We then show how to apply the metric to larger DPOP problems. Lastly, we briefly discuss applying our metric to another popular DCOP algorithm: Adopt.

4.1 Problem Set Up

We consider a a simple collaborative scheduling scenario with three people planning two meetings. We assume our adversary puts only negligible effort into discovering private information, looking at the messages and perhaps scribbling on a napkin. The

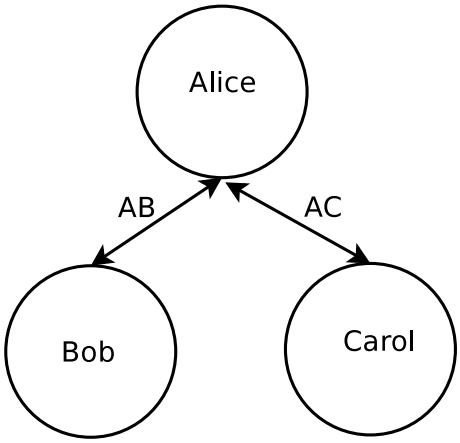

Figure 2: A simple collaborative scheduling example. Alice, Bob and Carol wish to optimally schedule two meetings: one between Alice and Bob (AB) and one between Alice and Carol (AC). They decide to use the DPOP algorithm and organize themselves into a tree rooted at Alice in order to do so.

meetings are between Alice and Bob (AB) and between Alice and Carol (AC). The constraint graph for this scenario is shown in Figure 2. The meetings can occur in any one of three timeslots. Each person has a valuation for each meeting and for each timeslot, each ranging from 0 to 5. The valuations for Alice, Bob and Carol are shown in Table 1. This is the private information in the scenario and there are 13 distinct pieces of it. This private information can be used to generate tables which give the valuation for all possible assignments of meetings to timeslots.

An agent's valuation for any proposed schedule can be calculated by summing the valuations for all scheduled meetings and subtracting the valuations for all scheduled timeslots. This is done for Alice, Bob and Carol in Table 2. Rows of this table are the typical type of information communicated in DCOP algorithms. We include possibilities for unscheduled meetings since in a highly constrained scheduling scenario, some meetings may be impossible to schedule or their elimination may lead to results with higher utility. Revealing an agent's table causes their private information in Table 1 to be easily derived. Revealing some subset of the rows may also allow some private information to be derived.

4.2 Applying the metric to DPOP

In our example, the agents are using DPOP [20] as their algorithm for solving the optimization problem. After describing the algorithm, we will show how DPOP performs according to our privacy metric.

DPOP is a synchronous dynamic programming algorithm that works through variable elimination. The agents are organized into a tree such that all agents who share constraints are on some path from root to leaf. Leaf nodes begin by sending their tables of valuations upward to their parents. The parents then aggregate their informa-

	Valuation
Meeting *AB*	4
Meeting *AC*	1
Timeslot 0	2
Timeslot 1	4
Timeslot 2	0

(a) Alice's Private Information

	Valuation
Meeting *AB*	4
Timeslot 0	1
Timeslot 1	0
Timeslot 2	5

(b) Bob's private information

	Valuation
Meeting *AC*	3
Timeslot 0	0
Timeslot 1	0
Timeslot 2	0

(c) Carol's private information

Table 1: Private information for Alice, Bob and Carol. There are 13 total pieces of private information, corresponding to the 13 rows in these three tables.

tion with that of their children, eliminating any unnecessary columns and finally send the result upwards. This continues until the root receives messages from its children, chooses the optimal values for its constraints and propagates the result downward. In this algorithm, leaf nodes expose all their private information to their parents. Internal nodes do not lose any valuation information (in the model where the adversary does not expend effort for inference) since their information is aggregated with their subtree, though they may expose some constraint graph information. In a perfect binary tree, just over half the nodes will be leaves. It is, of course, possible to build a DPOP tree that is less balanced and preserves more privacy, but efficiency will be sacrificed.

In our simple DPOP tree in Figure 2, Bob and Carol send their valuation tables (Table 2) to Alice, who combines them with her own valuation table to determine the optimal solution: meeting AB gets scheduled at timeslot 0 and meeting AC gets scheduled at timeslot 2. This turns out to be equivalent to a centralized algorithm, because one agent has all of the information at the end (Alice). All of Bob and Carol's information sum to eight pieces of personal information out of a total of 13. As a result the privacy loss is $\frac{8}{13}$ or 62%.

Varying the topology: In the previous example, we saw that using DPOP didn't reduce the privacy loss seen in a centralized algorithm. However, we suggested that a less balanced tree might lead to improvements. Consider the chain topology in Figure 3. Here only Carol sends up her table to Alice. Alice then aggregates her values with Carol's and sends a new table with valuations of her subtree to Bob, shown in Table 3. In this scenario, Bob cannot learn anything definitive about Alice or Carol, so the only information released is Carol's private information. This represents four pieces of

Timeslot 0	Timeslot 1	Timeslot 2	Alice's Valuation
-	-	-	0
AB	-	-	2
-	AB	-	0
-	-	AB	4
AC	-	-	-1
-	AC	-	-3
-	-	AC	1
AB	-	AC	3
AB	AC	-	-1
AC	-	AB	3
AC	AB	-	-1
-	AB	AC	1
-	AC	AB	1

(a) Alice's valuation table

Timeslot 0	Timeslot 1	Timeslot 2	Bob's Valuation
-	-	-	0
AB	-	-	3
-	AB	-	4
-	-	AB	-1

(b) Bob's valuation table

Timeslot 0	Timeslot 1	Timeslot 2	Carol's Valuation
-	-	-	0
AC	-	-	3
-	AC	-	3
-	-	AC	3

(c) Carol's valuation table

Table 2: Valuations for each possible assignment of meetings to timeslots. From each table, the agent's private information can be inferred.

private information so in our metric $D = \frac{4}{13}$ or 31% privacy loss.

Discussion: Revealing the stark difference that this change in topology causes is a feature of our metric. With a probabilistic metric, such as one based on entropy, this difference may be obscurred by the fact that Bob will likely be able to reduce the possible states for Alice and Carol's private information greatly from the large number that he had at the start of the algorithm. On the other hand, he still doesn't definitively know anything harmful about them. Averaging the privacy loss between all pairs of

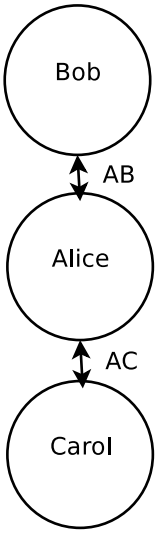

Figure 3: The same scenario as in Figure 2, but this time the "tree" is actually a chain with Bob at the root.

Timeslot 0	Timeslot 1	Timeslot 2	Valuation
-	-	-	4
AB	-	-	6
-	AB	-	4
-	-	AB	6

Table 3: Aggregated Valuations for Alice and Carol

agents would have caused the meaning of the metric to be further muddled.

Of course, not all DPOP problems are so small. A larger DPOP tree is illustrated in Figure 4. This tree schedules 14 meetings for 14 participants. There are 5 timeslots. Since each agent has private information for each meeting value and each timeslot, there are 99 total pieces of private information in the system : $14 * 5 = 70$ valuations for the timeslots and 29 meeting values (as can be seen by adding up the letters on the edges in Figure 4). The leaves lose all their information: 9 meeting valuations and 35 timeslot valuations. So we have approximately 44% privacy loss, by our metric.

No one-number metric will be able to completely capture everything we want to say about privacy in these algorithms, but we maintain that our proposed metric captures something meaningful that can be of assistance when deciding between algorithms, tuning their parameters and designing better algorithms.

4.3 Applying the metric to Adopt

Now, we turn to another prominent DCOP algorithm. Adopt [5] is an asynchronous algorithm that builds a tree with the same properties as DPOP. Instead of communi-

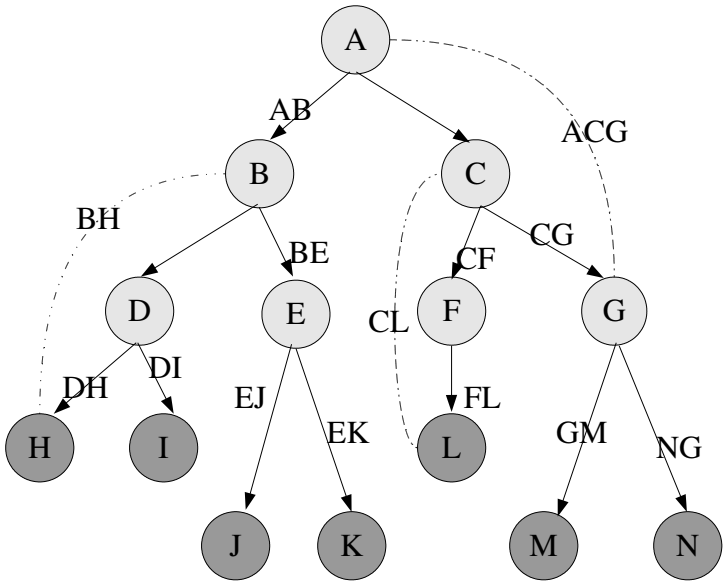

Figure 4: A larger DPOP tree. Solid lines represent avenues of communication for the algorithm. Meetings are labelled on the edge between the shallowest and deepest participant.

cating the whole table at once, results are sought one valuation at a time. A possible assignment of values is sent from parent to child, and the child replies with a valuation for that assignment. The child then propagates the value to its children. When the children reply, the agent sends an updated cost for its subtree to its parent. In this case, the whole table is never sent unless needed for optimization. However, every agent is able to elicit valuations for specific rows from its immediate children. In most cases, this information will be sufficient to uncover the child's personal information.

If we consider the example scenario described in Section 4.1, but use the Adopt algorithm rather than DPOP, an adversary might learn only a subset of the rows of Alice's valuation table, as shown in Table 4.

Though only part of Alice's valuation table in Table 2, the rows in Table 4 allow all of Alice's private information to be inferred. Alice has five pieces of private information: AB, AC, t_0, t_1, t_2. Each row of the table can be made into an equation:

Timeslot 0	Timeslot 1	Timeslot 2	Alice's Valuation
AC	-	-	-1
-	AC	-	-3
-	-	AC	1
AB	-	AC	3
AB	AC	-	-1

Table 4: A subset of Alice's valuation table. All of Alice's private information can be inferred.

Timeslot 0	Timeslot 1	Timeslot 2	Alice's Valuation
AB	-	AC	3
AB	AC	-	-1

Table 5: A smaller subset of Alice's valuation table. Only some information (i.e., the relative valuation of T_0 and T_2) can be inferred.

$$\begin{aligned} AC - t_0 &= -1 \\ AC - t_1 &= -3 \\ AC - t_2 &= 1 \\ AB + AC - t_0 - t_2 &= 3 \\ AB + AC - t_0 - t_1 &= -1 \end{aligned}$$

This system of equations can then be used to quickly deduce the ordering of the values:

$$((t_2 + 1 = AC) + 1 = t_0) + 2 = t_1 = AB$$

Then we set $t_2 = 0$ and learn all the values in the table ($AC = 1$, $t_0 = 2$, $t_1 = 4$, $AB = 4$).

In contrast, if we only have the last two lines/equations (Table 5), all we can infer is that $t_2 + 4 = t_1$, so when we normalize the least valued element to 0, we determine that $t_2 = 0$ and $t_1 = 4$, but not the values for the rest of Alice's private information.

Discussion: In both algorithms above, and also in other DCOP algorithms such as Synchronous Iterative Deepening (SynchID) and Synchronous Branch and Bound (SynchBB), the topology of the agents has large effects on the privacy of the algorithm. In addition, agents near the top of the trees in these algorithms leak less information than agents near the bottom. It is possible that these inequities might lead to resentment and undermine collaboration. On the other hand, these differences might be able to be exploited as the degree to which individuals value privacy varies widely. In addition, if agents share constraints with other, trusted agents, they might be able to structure their topology so that their information will only be available to an agent they trust, who will then aggregate the information with their other trusted agents.

5 RelatedWork

Initial work on privacy in constraint reasoning focused on the distributed constraint satisfaction problem (DisCSP). The goal of DisCSP is to find a solution that satisfies some set of hard constraints. A number of metrics were developed to demonstrate the privacy of DisCSP algorithms [9–11].

Maheswaran et al. proposed the Valuations of Possible States (VPS) framework to describe metrics for privacy loss in distributed constraint reasoning [7, 8]. They extended some metrics intended for DisCSP to work for DCOP. These metrics were the probabilistic metrics explained in Section 3, which lose too much meaning in aggregate. Maheswaran et al. used these metrics to experimentally show that some algorithms, which were assumed to protect privacy, lost more privacy than a centralized approach.

There has also been work on incorporating cryptographic techniques into DCOP and DisCSP algorithms in an attempt to avoid information leakage and thus eliminate privacy loss [18, 19]. Though this approach is appealing because of both the strong guarantees it can provide and the confidence it can instill in the users of such a system, it can also require the use of trusted servers or computationally intensive secure function evaluation techniques that may not always be available or justifiable. If we can bound privacy loss in specific DCOP algorithms using meaningful metrics, then cryptographic techniques may be avoidable, especially in situations where they are impractical.

6 Conclusion

There are clear privacy concerns in algorithms for collaborative scheduling. Previous metrics for privacy loss in these algorithms aggregate too much uncertain information, fail to capture intuitive notions of privacy, and do not view privacy loss in the context of actual threats.

We proposed $D|A$, a measure of the definitive private information revealed by an algorithm given an economic model of an adversary. We demonstrated the use of our metric, for a low-effort adversary, on scheduling scenarios for two popular DCOP algorithms.

We are working to experimentally evaluate our metric using a larger number of scenarios and algorithms. We are also interested in using our metric to determine which algorithms perform best when what is considered private changes. For instance, collaborative scheduling of secret meetings requires that the algorithms do not leak the entire constraint graph. Lastly, there is more work to be done on the A side of the metric, in quantifying the effort of adversaries.

7 Acknowledgments

Thanks to Milind Tambe, Jonathan Pearce, Emma Bowring, and Barbara Grosz for their helpful suggestions and support.

This work has been funded in part by NSF Trusted Computing grant CCR-0310877 and by gifts from Microsoft. Rachel Greenstadt was supported in part by a U.S. Department of Homeland Security (DHS) Fellowship, a program administered by the Oak Ridge Institute for Science and Education (ORISE). ORISE is managed by Oak Ridge Associated Universities under DOE contract number DE-AC05-00OR22750.

References

[1] Modi, P.J., Veloso, M.: Bumping strategies for the multiagent agreement problem. In: AAMAS. (2005)

[2] Hassine, A.B., Defago, X., Ho, T.: Agent-based approach ot dynamic meeting scheduling problems. In: AAMAS. (2004)

[3] Maheswaran, R.T., Tambe, M., Bowring, E., Pearce, J.P., Varakantham, P.: Taking DCOP to the real world: efficient complete solutions for distributed multi-event scheduling. In: AAMAS. (2004)

[4] Wagner, T.: Coordinators mission (2005) http://www.darpa.mil/IPTO/programs/coordinators/index.htm.

[5] Modi, P.J., Shen, W., Tambe, M., Yokoo, M.: ADOPT: Asynchronous distributed constraint optimization with quality guarantees. Artificial Intelligence Journal **161** (2005) 149–180

[6] Silaghi, M.C., Faltings, B.: A comparison of distributed constraint satisfaction approaches with respect to privacy. In: DCR. (2002)

[7] Maheswaran, R.T., Pearce, J.P., Varakantham, P., Bowring, E., Tambe, M.: Valuations of possible states (VPS):a quantitative framework for analysis of privacy loss among collaborative personal assistant agents. In: AAMAS. (2005)

[8] Maheswaran, R.T., Pearce, J.P., Bowring, E., Varakantham, P., Tambe, M.: Privacy loss in distributed constraint reasoning: A quantitative framework for analysis and its applications. JAAMAS (2006)

[9] Franzin, M.S., F.Rossi, Freuder, E.C., Wallace, R.: Multi-agent meeting scheduling with preferences: efficiency, privacy loss, and solution quality. Computational Intelligence **20**(2) (2004)

[10] Silaghi, M.: Meeting scheduling guaranteeing n/2-privacy and resistant to statistical analysis (applicable to any discsp). In: 3rd IC on Web Intelligence. (2004)

[11] Meisels, A., Lavee, O.: Using additional information in DisCSPs search. In: DCR. (2004)

[12] Acquisti, A., Grossklags, J.: Privacy and rationality in decision making. IEEE Security and Privacy (2005)

[13] Yokoo, M., Durfee, E.H., Ishida, T., Kuwabara, K.: The distributed constraint satisfaction problem: formalization and algorithms. IEEE Transactions on Knowledge and Data Engineering **10**(5) (1998)

[14] Greenstadt, R., Pearce, J., Bowring, E., Tambe, M.: An experimental analysis of privacy loss in dcop algorithms. In: DCR. (2006)

[15] Ellsberg, D.: Risk, ambiguity and the savage axioms. Quarterly Journal of Economics **75** (1961)

[16] Sweeney, L.: k-anonymity: a model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems **10**(5) (2002)

[17] Schechter, S.: Computer Security Strength and Risk: A Quantitative Approach. PhD thesis, Harvard University (2004)

[18] Yokoo, M., Suzuki, K., Hirayama, K.: Secure distributed constraint satisfaction: Reaching agreement without revealing private information. In: CP 2002, Berlin (2002)

[19] Silaghi, M.C., Mitra, D.: Distributed constraint satisfaction and optimization with privacy enforcement. In: IAT. (2004)

[20] Petcu, A., Faltings, B.: A scalable method for multiagent constraint optimization. In: IJCAI. (2005)