

# **EXPERIMENTAL ANALYSIS OF PRIVACY LOSS IN DCOP ALGORITHMS**

**Greenstadt, R, K.**

CREATE REPORT  
Under FEMA Grant EMW-2004-GR-0112

**Aug 1, 2005**



**Center for Risk and Economic Analysis of Terrorism Events  
University of Southern California  
Los Angeles, California**

# Experimental analysis of privacy loss in DCOP algorithms

Rachel Greenstadt (student)  
Harvard University  
greenie@eecs.harvard.edu

Jonathan P. Pearce, Emma Bowring and  
Milind Tambe  
University of Southern California  
{jppearce,bowring,tambe}@usc.edu

## ABSTRACT

Distributed Constraint Optimization (DCOP) is rapidly emerging as a prominent technique for multiagent coordination. Unfortunately, rigorous quantitative evaluations of privacy loss in DCOP algorithms have been lacking despite the fact that agent privacy is a key motivation for applying DCOPs in many applications. Recently, Maheswaran et al. [5] introduced a framework for quantitative evaluations of privacy in DCOP algorithms, showing that early DCOP algorithms lose more privacy than purely centralized approaches and questioning the motivation for applying DCOPs. Do state-of-the-art DCOP algorithms suffer from a similar shortcoming? This paper answers that question by investigating several of the most efficient DCOP algorithms, including both DPOP and ADOPT. Furthermore, while previous work investigated the impact on efficiency of distributed constraint reasoning design decisions, e.g. constraint-graph topology, asynchrony, message-contents, this paper examines the privacy aspect of such decisions, providing an improved understanding of privacy-efficiency tradeoffs. Finally, this paper augments previous work on system-wide privacy loss, by investigating inequities in individual agents' privacy loss.

## 1. INTRODUCTION

Personal assistant agents are an emerging tool for collaboration in businesses, office environments and research organizations [3, 6, 10]. To perform their function, these agents must be endowed with potentially private information about their users, e.g. salary, capabilities, and preference information about meetings and schedules. In the course of negotiations and conflict resolutions, the exchange of some private information is necessary to achieve a good team outcome. For humans to entrust their personal assistant agents with private information, they need assurance that their privacy will be protected. Thus, understanding how such applications lose privacy in multiagent negotiations is crucial for their success.

Promising approaches in distributed constraint reasoning (DCR), such as distributed constraint satisfaction (DisCSP) [14, 15] and distributed constraint optimization (DCOP) [6, 7, 9], enable distributed conflict resolution and coordination while maintaining users' privacy. Indeed, maintaining privacy is a fundamental motivation in DCOP [6, 9, 13, 15], and thus DCOP is now heavily applied in software personal assistants [1, 3, 6, 10, 13]. One approach to privacy

in DCOP is to use cryptographic techniques [16] that ensure watertight privacy but require the use of external servers or computationally intensive cryptographic operations that may not always be available or justifiable for their benefits. Instead, we focus on a second approach in which researchers provide metrics for quantifying the privacy loss in DCOP algorithms [2, 5, 8, 13]. If we can bound privacy loss in specific DCOP algorithms, then cryptographic techniques may be avoidable in situations where they are impractical.

Unfortunately, there are three key weaknesses in the previous work on privacy loss analysis in DCOP. First, recent cross-algorithm privacy loss analysis focused on a limited number of DCOP algorithms but indicated that these algorithms preserve less privacy than a centralized approach [5], seriously undermining a key motivation for these algorithms. Thus, given the importance of privacy in motivating applications of DCOPs, it is crucial to analyze some of the most used and most recent DCOP algorithms to see whether they are similarly undermined and to measure their cross-algorithm performance. Two notable omissions in previous analysis are ADOPT [9] and DPOP [11], both among the most efficient DCOP algorithms that provide very disparate points in the design space of DCOP algorithms. Second, while the impact of DCOP design decisions on efficiency has been investigated — in particular, constraint-graph topology, asynchrony, and message content — the impact on privacy has not. This tradeoff between privacy and efficiency is critical to DCOP algorithms. Third, while previous investigations have examined system-wide privacy loss, it is also important to investigate inequities in privacy loss, e.g. does one agent accumulate more information than others?

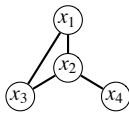
This paper overturns the significant negative results from [5] by providing positive privacy results for several state-of-the-art DCOP algorithms not considered in [5]. This paper (i) analyzes ADOPT, DPOP and SynchID [9], three recent DCOP algorithms; (ii) analyzes the privacy impact of DCR design decisions, including constraint-graph topology, asynchrony and message-contents; and (iii) analyzes disparities in privacy loss among individual agents. These contributions are obtained by a large-scale experimental investigation of privacy loss in DCOP algorithms in the VPS (Valuations of Possible States) analysis framework [5], using several distributed meeting scheduling scenarios. To further understand privacy loss in DCOP, we also investigate upper bounds on privacy loss in DCOP algorithms. Overall, while our results are more promising than [5], our upper bound results indicate the need for further attention to privacy preservation in DCOP algorithms.

## 2. BACKGROUND

A DCOP consists of a set of variables assigned to agents who control their values. The agents must coordinate their local choice of variable values so that a global objective function, modeled as a set of distributed valued constraints, is optimized. Figure 1(a)

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...\$5.00.



$d_i$	$d_j$	$f(d_i, d_j)$
0	0	1
0	1	2
1	0	2
1	1	0

Figure 1: DCOP

shows an example DCOP with four agents. Each constraint is defined by the table shown, where  $f(d_i, d_j)$  denotes the cost of assigning  $x_i = d_i$  and  $x_j = d_j$ . The objective is to find an assignment  $\mathcal{A}$  such that the total cost, denoted  $F$ , is minimized:

$$F(\mathcal{A}) = \sum_{x_i, x_j} f_{ij}(d_i, d_j), \text{ where } x_i \leftarrow d_i, x_j \leftarrow d_j \in \mathcal{A} \quad (1)$$

Thus in Figure 1,  $x_1, x_2, x_3$ , and  $x_4$  are variables each with domain  $\{0, 1\}$ , and the optimal assignment is  $(x_1, 1), (x_2, 1), (x_3, 1), (x_4, 1)$ .

Given the recent interest in DCOP, several competing algorithms for DCOPs have now been proposed, including Adopt [9], SynchID [9], DPOP [11], and OptAPO [7]. In addition, SynchBB [4] is an early algorithm for DCOP. Previous work has provided a comparison of privacy loss of a centralized approach with OptAPO and SynchBB, suggesting that the centralized approach may lead to lower privacy loss. Hence this paper focuses on the remaining algorithms above. These algorithms also present novel design choices, or occupy a prominent place in the algorithmic space. The following describes key characteristics of these remaining algorithms:

**Adopt** is an asynchronous complete DCOP algorithm, guaranteed to find the optimal solution. In Adopt, an agent communicates only one value from its domain at a time, or one cost message (indicating the cost of an assignment to some set of variables) at a time. Such restricted communication is intended to both limit the size each message, and preserve privacy. Additionally, Adopt’s asynchrony, combined with several heuristic techniques, provides significant speedups over synchronous approaches. A key feature of Adopt is that the DCOP graph is converted into a Depth-First Search (DFS) tree in which constraints can exist between a variable and any of its ancestors or descendants, but not between variables in separate sub-trees (e.g. see Figure 1). Such a graph allows increased parallelism in the search. Given such a DFS tree, Adopt uses a search strategy similar to iterative deepening search. As a leading DCOP algorithm, Adopt provides an interesting case study. Its asynchrony and restricted communication are also of interest.

**SynchID** is an iterative deepening algorithm similar to Adopt, with two primary differences. First, SynchID requires that variables be ordered in a linear chain, and thus cannot take advantage of the parallelism of a tree structure, as Adopt can. Second, SynchID is a synchronous algorithm, in which messages are sent only at pre-determined intervals, with agents executing sequentially according to the chain ordering, whereas in Adopt, agents send messages concurrently and asynchronously. Therefore, SynchID usually requires more message cycles (and time) than Adopt, but allows for fewer actual messages to be sent. The reduced number of messages was expected to provide significant additional privacy in SynchID, thus illustrating that Adopt’s pursuit of efficiency via asynchrony is detrimental to privacy. Thus, SynchID is included in our set of algorithms to allow comparison with Adopt to understand the impact of asynchrony on privacy loss.

**DPOP** [11] is a synchronous complete DCOP algorithm. Also, one of the leading DCOP algorithms, DPOP also requires the creation of a DFS tree similar to Adopt. However, a key difference in DPOP is that agents communicate to their parents the cost of their subtree’s best assignment for every value in the parent’s domain (or every combination of parent and pseudo-parents’ values). A vari-

able elimination based algorithm, DPOP is able to synchronously communicate significantly fewer messages to solve DCOPs (at least when compared to Adopt), but at the cost of significantly larger message size. The expectation was that due to DPOP’s large amount of information per message, privacy loss would be significant, making DPOP a prime candidate for comparison with other algorithms.

**SynchBB** or synchronous branch-and-bound was studied in [5]. However, we focus on a slightly modified SynchBB where some key useless information is not communicated. In particular, the original SynchBB organized agents in a chain, and communicated all the value assignments received from its parents downstream, our modified version only communicates relevant value assignments, in the interest of privacy. SynchBB, a synchronous algorithm that sends cost messages up and down a chain, provides a fourth point of comparison and a measure of the impact of message directions on privacy loss.

### 3. EXPERIMENTAL METHODOLOGY

Given the applicability of DCOP to the personal assistant agent domain (specifically distributed meeting scheduling) [1, 10] as well as the privacy concerns inherent in this domain [2, 5], we focus our investigation specifically on privacy loss in a DCOP formulation for the distributed meeting scheduling problem. However, the results of this work can be generalized to other problems in which cooperative agents have privacy concerns.

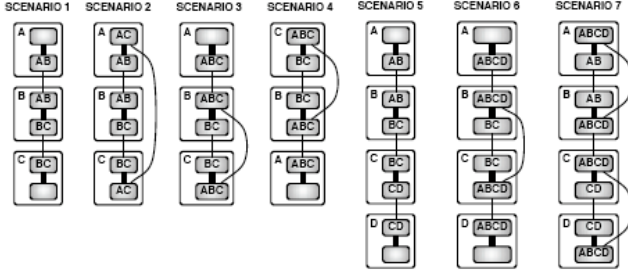
#### 3.1 Problem Formulation

We define a meeting/event scheduling problem based on the formalism of [6] as follows:

- $\mathcal{R} := \{R_1, \dots, R_N\}$  is a set of  $N$  agents (representing people).
- $\mathcal{E} := \{E^1, \dots, E^K\}$  is a set of  $K$  events.
- $\mathcal{T} := \{1, \dots, T\}$  is the set of available timeslots.
- $E^k := (A^k, V^k)$  is the  $k^{\text{th}}$  event, where  $A^k \subset \mathcal{R}$  are the people required to attend the event and  $V^k := \{V_1^k, \dots, V_N^k\}$  is a value vector, where  $V_n^k$  represents the value to the  $n^{\text{th}}$  person for scheduling event  $k$ .
- $V_n^0(t) : \mathcal{T} \rightarrow \mathbb{R}^+$  denotes the  $n^{\text{th}}$  person’s valuation for keeping time slot  $t$  free. This might be due to a preference to keep that time open or the value of an already scheduled event.

We define a schedule  $S$  as a mapping from the event set to the time domain where  $S(E^k) \in \mathcal{T}$  denotes the time slot committed for event  $k$ . This implies that all people in  $A^k$  must agree to assign the time slot  $S(E^k)$  to event  $E^k$  in order for the event to be *scheduled*, allowing them to obtain the utility for attending it. A scheduling conflict occurs if two events with at least one common required attendee are scheduled at the same time. An assignment of  $S(E^k) = \emptyset$  implies that event  $E^k$  is not scheduled. We define a person’s utility to be the difference between the sum of the values from scheduled events and the aggregated values of the time slots used for those scheduled events. The team wants to maximize the sum of utilities of all its members in order to best utilize their assets.

To explore the privacy loss in DCOP algorithms for meeting scheduling, we express the problem using the PEAV-DCOP representation [6], which is motivated by privacy considerations. In a DCOP, each agent controls a set of variables. In PEAV-DCOP, each of these variables in an agent’s set corresponds to an event (meeting) that agent’s user must attend; i.e. each event is represented by a set of variables, each controlled by one of that event’s attendees.



**Figure 2: Scenarios:** Transparent boxes represent agents and the dark, inner boxes are meeting variables. Thick lines are intra-agent constraints and thin lines are inter-agent constraints.

This allows us to design constraint utility functions where all valuation information is on internal links, thus maintaining privacy. The DCOP is expressed as follows:

- For each event  $E^k, X^k := \{x_n^k : R_n \in A^k\}$  is a set of variables where  $x_n^k \in \{0, 1, \dots, T\}$  denotes the starting time for event  $E^k$  in the schedule of  $R_n$ , a required person for  $E^k$ .
- $X := \cup_{k=1}^K X^k$  is thus the complete set of variables in the DCOP. Each variable's domain is the set of timeslots  $\mathcal{T}$ .
- For each agent,  $R_n, \tilde{X}_n := \{x_n^k \in X\} \subset X$  is the set of variables controlled by  $R_n$ , where each variable represents an event. If  $|\tilde{X}_n| = 1$ , let  $X_n := \tilde{X}_n \cup \{x_n^0\}$  where  $x_n^0 \equiv 0$  is a dummy variable, in order to ensure that intra-agent constraints exist for all agents. Otherwise,  $X_n := \tilde{X}_n$ .
- For each event  $E^k$ , inter-agent constraints exist between all pairs of variables  $x_m^k, x_n^k$  in  $X^k$ , such that  $f(t_m^k, t_n^k) = -\infty$  if  $t_m^k \neq t_n^k$  and  $f(t_m^k, t_n^k) = 0$  otherwise. (We use  $t_n^k$  to mean the timeslot chosen by agent  $R_n$  for event  $E^k$ ). This is to ensure that all participants in an event agree on its time (or agree that it will not be scheduled).
- For each agent,  $R_n$ , intra-agent constraints exist between all pairs of variables  $x_n^j, x_n^k$ , such that the sum of rewards on all intra-agent constraints for  $R_n$  equals the net gain to  $R_n$ , i.e.  $\sum V_n^k - V_n^0(t_n^k)$  if no two variables have the same value  $t > 0$ . Otherwise, this sum should equal  $-\infty$  to avoid double-booking any time slots. The mapping of costs and rewards onto individual constraints could be done many ways, such as splitting the net reward for each meeting evenly among all outgoing intra-agent constraints from that meeting.

### 3.2 Scenarios in PEAV-DCOP

The majority of scheduling instances in a functional personal assistant agent system will consist of a small number of meetings that need to be negotiated simultaneously. This notion of a small number of meetings is also shared by [4, 11] and as members of a research organization, this is a situation that the authors commonly observe. While larger-scale problems may present themselves, if privacy is a critical factor, the coordination protocols must be effective for these small-scale instances. We consider seven scenarios of three ( $R = \{A, B, C\}$ ) or four ( $R = \{A, B, C, D\}$ ) agents. The PEAV-DCOP graphs in Figure 2 show the events, labeled by their attendees, and decomposed into variables and constraints.

### 3.3 VPS: Measuring Privacy Loss

The Valuation of Possible States (VPS) framework [5] was proposed to quantitatively evaluate privacy loss in multiagent settings.

Quantification of privacy loss in VPS is based on a valuation on the other agents' estimates about (i.e. a probability distribution over) an agent's possible states. There are three key elements in VPS: (i) agent  $R_n$ 's private information, modeled as a state  $s_n \in S_n$ , where  $S_n$  is a set of possible states that  $R_n$  may occupy; (ii) other agents' estimates about agent  $R_n$ 's possible states, expressed as a probability distribution  $\mathbb{P}_n((S_n)^{N-1})$ , (iii) the utility that agent  $R_n$  derives from the distribution of other agents' beliefs about  $R_n$ 's states, yielding value function  $\mathbb{V}_n(\mathbb{P}_n((S_n)^{N-1}))$ . Note that  $\mathbb{P}_n((S_n)^{N-1}) = [\mathbb{P}_n^1(S_n) \mathbb{P}_n^2(S_n) \dots \mathbb{P}_n^N(S_n)]$ , where  $\mathbb{P}_n^j(S_n)$  provides agent  $R_j$ 's probability distribution over states of agent  $R_n$ .

In applying VPS to our meeting scheduling scenario, encoded as a PEAV-DCOP, the initial task is to identify the information that an agent should consider private, i.e. the data that identifies the state of its human user. In our meeting scheduling problem, it is clear that the valuation of time,  $V_n^0(t)$ , explicitly captures the preferences that will be used in the collaborative process. In addition, the rewards for attending various events  $\{V_n^k : R_n \in A^k\}$  are another component that agents may wish to keep private. For the sake of simplicity in analysis, we will assume we are in a setting where event rewards are public information, though the analysis can be extended to capture situations where this information is private as well. If  $V_n^0(t) \in \mathcal{V}$  where  $\mathcal{V}$  is a discrete set and there are  $T$  time slots in a schedule, the state  $w_n$  of the  $n^{\text{th}}$  agent is an element of the set  $S_n = \mathcal{V}^T$  and can be expressed as a vector of length  $T$ . Before negotiation, each agent knows only that the other agents exist in one of  $|\mathcal{V}|^T$  possible states. After negotiation, each agent will be modeled by all other agents whose estimate of the observed agent is captured by  $\mathbb{P}_n((S_n)^{N-1})$ . One question is how an agent should assign values to these estimates of possible states through which others see it. Different possibilities for how these values may be assigned are captured in six metrics introduced in [5] that define the privacy of a single agent with respect to all others. We list them briefly below.

Due to the nature of messaging in DCOPs, the typical form of information gathered is the elimination of a possible state. ProportionalS gives the number of states not eliminated by other agents:

$$\mathbb{V}_n(\mathbb{P}_n((S_n)^{N-1})) = \sum_{j \neq n} \sum_{s_n \in S_n} I_{\{\mathbb{P}_n^j(s_n) > 0\}} \quad (2)$$

where  $I_{\{\cdot\}}$  is an indicator function.

The GuessS metric models privacy as the sum of probabilities that each other agent will be unable to guess the observed agent's state accurately, given that their guesses are chosen uniformly over their set of possible states for the observed agent:

$$\mathbb{V}_n(\mathbb{P}_n((S_n)^{N-1})) = \sum_{j \neq n} \left( 1 - \frac{1}{\sum_{s \in S_n} I_{\{\mathbb{P}_n^j(s) > 0\}}} \right) \quad (3)$$

EntropyS was introduced in [2] and considers privacy loss from an information-theoretic perspective:

$$\mathbb{V}_n(\mathbb{P}_n((S_n)^{N-1})) := \log_2 \left( \frac{\sum_{j=1}^{|\mathcal{V}|^T} I_{\{\mathbb{P}_n^G(s_j) > 0\}}}{|\mathcal{V}|^T} \right) \quad (4)$$

where  $G = \mathcal{R} \setminus R_n$ .

Each of the metrics has an analogous metric below, representing privacy loss on a per-timeslot basis, averaged over all timeslots.

ProportionalTS:

$$\mathbb{V}_n(\mathbb{P}_n((S_n)^{N-1})) = \sum_{j \neq n} \sum_{k=1}^T \sum_{m=1}^{|\mathcal{V}|} I_{\{\max_{s_n \in S_n: s_n(k)=m} \mathbb{P}_n^j(s_n(k)=m) > 0\}} \quad (5)$$

GuessTS:

$$\mathbb{V}_n(\mathbb{P}_n((S_n)^{N-1})) = \sum_{j \neq n} \sum_{k=1}^T \left( 1 - \frac{1}{\sum_{m=1}^{|\mathcal{V}|} I_{\{\max_{s_n \in S_n: s_n(k)=m} \mathbb{P}_n^j(s_n(k)=m) > 0\}}} \right). \quad (6)$$

EntropyTS:

$$\mathbb{V}_n(\mathbb{P}_n((S_n)^{N-1})) := \sum_{j \neq n} \sum_{k=1}^T \log_2 \left( \frac{\sum_{m=1}^{|\mathcal{V}|} I_{\{\max_{s_n \in S_n: s_n(k)=m} \mathbb{P}_n^j(s_n(k)=m) > 0\}}}{|\mathcal{V}|} \right) \quad (7)$$

We can scale all functions with a transformation of the form  $\tilde{\mathbb{V}} = \alpha(\mathbb{V} - \beta)$  with appropriate choices of  $\alpha$  and  $\beta$  such that the valuations span  $[0, 1]$  with zero being the worst level and one being the ideal level of privacy. We can then subtract these scaled functions from one to show an agent's *privacy loss*, for which a value of zero means that no privacy is lost and a value of one means that all private states are known to all other agents.

### 3.4 Inference Algorithms

Based on the VPS framework, and the privacy metrics given above, we define a process by which agents can infer information about other agents while running various DCOP algorithms, in order to measure the likely privacy loss between agents in a DCOP. All inference experiments for all algorithms (including centralized) start with the same initial assumptions. We assume that the constraint graph and the value (reward) of each meeting is known to all agents, but the valuations of time slots are private. These assumptions are exactly as in [5], allowing a comparison of the two results; in addition, for the scenarios with few meetings, it is reasonable to assume that the rewards for meetings are public knowledge. Based on these assumptions, we developed the following methods for agent inference for SynchronID, Adopt and DPOP.

**Centralized:** In a centralized algorithm, the agents all send their valuation information to one agent, who computes the result and returns. In every case the centralized agent can “infer” the valuations perfectly [5]. Since we express our results as an average of each agent's privacy loss, the privacy loss of the centralized algorithm is  $\frac{1}{N}$  (for all six metrics) where  $N$  is the total number of agents.

**SynchronID:** SynchronID is a synchronous algorithm in which agents are ordered in a chain, and messages are passed up and down the chain. An upward message contains a number  $m_n$ , which is equal to the best currently known total reward for the chain. For PEAV, the total reward for the chain is equal to the sum of differences between the valuation of a scheduled meeting and the valuation of the time slot it occupies for every scheduled meeting for every person. We henceforth use  $\Delta_{R_n}^{E_k}(t) = V_n^k - V_n^0(t)$  to denote the change in utility to the  $n^{\text{th}}$  agent for scheduling the  $k^{\text{th}}$  event at time  $t$ . When agent  $R_n$  receives an upward message it knows that  $m_n$  is a sum of  $\Delta$ s downstream from  $R_n$ .

To illustrate how possible states can be eliminated in SynchronID, we outline the inferences that one can make from messages received in Scenario 1. In SynchronID, upward messages to agent  $R_n$  contained information of the form:

$$m_n = \sum \Delta_{R_n}^{E_k}(t_{E_k}) + \sum \Delta_{R_n}^{E_k}(\tilde{t}_{E_k}), \quad (8)$$

where the summations include events downstream from  $R_n$ .  $t_{E_k}$  is the time of an event  $E_k$  when that time is known to  $R_n$  (because  $R_n$  is a participant in event  $E_k$ ), and  $\tilde{t}_{E_k}$  is the time of an event  $E_k$  when that time is not known to  $R_n$ . For example, since  $B$  knows when meeting  $BC$  is scheduled, as well as the value of meeting  $BC$ , a message from  $C$  to  $B$  ( $m_B$ ) allows  $B$  to know  $V_C(t_{BC})$  (the valuation vector component of  $C$  at the time at which meeting  $BC$  is scheduled). Similarly, a message from  $B$  to  $A$  ( $m_A$ ) allows  $A$  to know  $v_B(t_{AB}) + v_B(\tilde{t}_{BC}) + v_C(\tilde{t}_{BC})$ , where  $\tilde{t}_{BC}$  is some time not equal

to  $t_{AB}$ , but otherwise unknown to  $A$ . Each one of these relationships allows the observing agent to reduce the number of possible states the other agents could be in. We obtain the privacy loss for SynchronID for the privacy metrics introduced earlier by allowing each agent to collect these relationships, and iterate over them, testing each relationship against a list of possible states for the other agents, discarding states that conflict with any of the relationships.

**Adopt:** Adopt contains the same type of upward messages as in SynchronID, but, due to its asynchrony, it may be impossible for agents to tell how many  $\Delta$ s are contained in the reward component of each message, information helpful for inference. When a message is received, we know it contains rewards for at least one agent more than the previous message it sent (in the case of SynchronID, it is always one agent, hence the = sign). However, due to asynchrony, our agent might have included more descendants in the message. So, for our inference, we use a  $\leq$  sign. The inference equation is:

$$m_n \leq \sum \Delta_{R_n}^{E_k}(t_{E_k}) + \sum \Delta_{R_n}^{E_k}(\tilde{t}_{E_k}), \quad (9)$$

This relation changes to an equality in the special case when only one agent is downstream from agent  $R_n$ .

**DPOP:** In the DPOP algorithm, each agent sends exactly one cost message to its parent. This message consists of a table of all possible assignments of constrained upstream events and the aggregate costs of those assignments to the agents downstream of  $R_n$ . Each entry in the table is used to create inference rules like those in SynchronID. The events in the entry are the  $\Delta_{R_n}^{E_k} t_{E_k}$  terms and other events with participating agents downstream of  $R_n$  are the  $\Delta_{R_n}^{E_k} \tilde{t}_{E_k}$  terms.

$$m_n = \sum \Delta_{R_n}^{E_k}(t_{E_k}) + \sum \Delta_{R_n}^{E_k}(\tilde{t}_{E_k}), \quad (10)$$

**SynchronBB:** Inference rules for SynchronBB are as described in previous work [5].

## 4. EXPERIMENTAL RESULTS

In this section, we present experimental results from the seven scenarios. We begin with comparisons of privacy loss in the studied algorithms according to the EntropyTS metric, then examine the new algorithms using all our metrics. We introduce a new metric to highlight privacy benefits of all studied DCOP algorithms over centralized approaches. We then explore the privacy impact of more sophisticated inference techniques and diverse topologies.

For the three-agent scenarios, we varied  $T$ , the number of timeslots, from 3 to 7 while holding  $|\mathcal{V}| = 3$ . Then, we varied  $|\mathcal{V}|$  from 3 to 7 while holding  $T = 3$ . For the four-agent scenarios, for reasons of computational complexity, we varied  $T$  from 3 to 4 while holding  $|\mathcal{V}| = 3$  and then varied  $|\mathcal{V}|$  from 3 to 5 while holding  $T = 3$  (Using  $T = 7$  in a four-agent scenario would require an agent to consider over a billion states:  $3^{(3 \cdot 7)}$ ). For each  $(T, |\mathcal{V}|)$  pair, we performed 10 runs for each of the following algorithms: SynchronID, Adopt, SynchronBB and DPOP. For each run, the privacy loss for each agent was measured using each of the six metrics, assuming the agents were using the inference algorithms given in Section 3. The systemwide privacy loss was expressed as the arithmetic mean of each agent's privacy loss:  $\sum_N \mathbb{V}_n/N$ .

Space limitations preclude us from presenting all our results. Thus, in some cases, we present results from only three of the seven scenarios. We also present results only on variations of valuations, although the results from variations in timeslots were similar (complete results available at <http://teamcore.usc.edu/dcop/privacy>). In each of our graphs, each data point is an average of 10 runs, and we provide statistical significance results to support our main conclusions. Finally, we use a chain topology for all algorithms, to allow

cross-algorithmic comparison (since not all algorithms could use a DFS tree as noted in Section 2), and to separate out the impact of graph topology on privacy in a separate experiment.

**Cross-algorithm comparison:** Figure 3 shows the comparison of privacy loss for the four algorithms mentioned above, for each of the seven scenarios, as well as providing a comparison of privacy loss with the centralized approach. The  $x$ -axis plots the different number of valuations (with number of time-slots fixed at 3) and the  $y$ -axis plots privacy loss. The thick horizontal line shows the centralized approach, for scenarios 1-4 (three agents), its privacy loss is 0.33, but for scenarios 5-7 (four agents) it is 0.25. The privacy loss in the centralized case is the same no matter which of the six metrics is used to measure it. We use the EntropyTS metric as the metric for privacy loss in this result; as seen later, EntropyTS provides results that are in the mid-range among all metrics. Nonetheless, we provide results for other metrics, and mostly these metrics agree with the conclusions drawn using the EntropyTS metric.

We conclude the following from Figure 3: (1) Except for SynchBB, the remaining algorithms have a privacy loss that is lower than the centralized approach. In contrast with the negative results presented in [5], which illustrated DCOP algorithms as having worse privacy loss than a centralized approach, this is a significant positive result. Indeed, the privacy loss in Adopt and DPOP is less than half that of the centralized approach. Furthermore, statistical tests show that Adopt performs better than centralized in all scenarios and DPOP performs better than centralized in all except scenario 4 (significance level of 5%). (2) DPOP and Adopt had very similar privacy loss, despite their vastly different approaches. In particular, despite DPOP’s one shot communication of all information, it performed surprisingly well in terms of privacy loss. Adopt does perform slightly better than DPOP for privacy loss (see in particular Scenario 4), but not to the level anticipated at least in these scenarios. (3) Adopt significantly outperformed SynchID in terms of privacy protection. The asynchrony in Adopt was expected to be significantly detrimental to privacy due to the increased numbers of messages. Instead, we found that the uncertainty introduced by asynchrony as to which agents participate in each cost message provides significant privacy gains compared to synchronous algorithms such as SynchID. (4) Despite modifications to improve privacy, SynchBB still performed the worst in terms of its privacy loss; often worse than centralized. The key reason for SynchBB’s low performance is its bi-directional messaging of cost information. Thus, it is important to avoid bi-directional cost propagation in DCOP algorithms when privacy is a goal.

**Cross-metric comparison:** We performed cross-metric comparisons for Adopt (Figure 4), SynchID (Figure 6), and DPOP (Figure 5), in which privacy loss was plotted for each of the six metrics from Section 3. Privacy loss for scenarios 1, 4, and 5 are plotted on the  $y$ -axis in each graph and the number of valuations is varied along the  $x$ -axis; the number of timeslots was fixed at three. In all cases, the metric indicating the highest privacy loss was LinearS, usually well above the others. Adopt outperformed the centralized approach according to all metrics except the LinearS metric, for which the centralized approach gave lower privacy loss in Scenario 4. SynchID also outperformed the centralized approach for all metrics except LinearS and except for Scenario 4, where the privacy loss according to both LinearS and GuessTS was 0.5. Similarly, DPOP outperformed the centralized approach for all metrics except LinearS and except for Scenario 4, where half the metrics placed it above the centralized approach, and half placed it below. However, for all three algorithms, at least half of the metrics showed them to outperform the centralized algorithm in every scenario, and in many scenarios, all of the metrics showed this to be true.

We conclude the following from Figure 4, Figure 5 and Figure 6: (1) Even if we examine other metrics beyond EntropyTS, DCOP algorithms do not suffer from privacy loss to the extent seen in the earlier investigation [5], further confirming the positive results seen earlier. (2) Metric-to-metric comparisons in general also appear to concur with the earlier conclusion about similarity of privacy loss in DPOP and ADOPT. However, LinearS and EntropyS do show a divergent trend between the two – with Adopt more significantly outperforming DPOP in terms of these metrics.

**MAX metric:** So far, all metrics measured the loss of privacy from one agent to another, which was then averaged to find the systemwide privacy loss. The effect of one agent learning more than others, and gaining an asymmetric advantage over them, is not considered. To address this issue, we devised the MAX metric. In this metric, we consider only the total privacy loss to the single agent that learns the most information about other agents (by EntropyTS), rather than the mean of the individual privacy loss figures. Figure 7 shows the results for all the algorithms according to the MAX metric for all seven scenarios, with  $T = 3$ . The number of valuations is plotted on the  $x$ -axis and the privacy loss is plotted on the  $y$ -axis.

We conclude the following from the MAX metric results: (1) The central agent in a centralized algorithm always learns all the other agents’ valuations. Thus, the centralized algorithm always gets a value of one according to the MAX metric. The MAX metric shows that there is always a privacy benefit obtained by using DCOP algorithms, even those that perform worse than centralized by our other metrics, if the major privacy concern is an advantage that can be gained by one agent accumulating knowledge. (2) Even with the MAX metric, DPOP and Adopt tended to outperform SynchBB, while SynchID varied widely from scenario to scenario.

**Upper bounds:** The results so far all used the inference algorithms described in Section 3.4, and provided a lower bound on privacy loss, since inference was done using only the contents of messages containing costs. Although lower bounds are sufficient to demonstrate a negative result, they must be augmented to demonstrate a positive one. While theoretically there is no limit to the quantity of domain knowledge an inference algorithm could bring to bear (making a true upper bound impossible to calculate), we can calculate upper bounds given knowledge only of the message contents and graph structure.

We calculated upper bounds on privacy loss for one of the most promising DCOP algorithms: DPOP. We used a brute force approach which generated all possible combinations of input valuations, ran DPOP on them to generate a trace of the messages each combination would produce and then for each agent matched these up to the messages that were actually received. We performed simulations of this type for DPOP, which took several days to run, compared to the several hours taken by our primary inference algorithms. Results for upper and lower bound inference for DPOP using all six metrics are shown in Figure 8. Scenarios 1, 4, and 5 are shown for three timeslots and three valuations, with the metric plotted on the  $x$  axis and the privacy loss plotted on the  $y$  axis. Due to asynchrony and the randomness in a variable’s initial choice of value, it is not possible to analyze ADOPT with this approach.

For each scenario, the lower bound showed DPOP outperforming the centralized approach (except on LinearS for Scenario 4) while the upper bound was comparable or worse than centralized. We conclude that while privacy results on recent DCOP algorithms are encouraging, there is still a need for improvement.

**Asynchrony:** The privacy loss of an asynchronous algorithm such as Adopt is difficult to analyze. Due to its asynchrony, it may be difficult for agents to ascertain which (or even how many) other agents’ valuations are part of any particular cost message.





