AdCSP — Cooperative Ad Auctions with Privacy Guarantees

Roussi Roussev and Marius Silaghi

Florida Institute of Technology

Abstract. This paper discusses a cooperative model for advertisement bidding with privacy guarantees. In our application, a set of participants need to agree on an allocation of advertisements to be shown on an Internet stream. The participants have preferences that pose constraints to what should be streamed, and define the value of the ads.
Streaming an ad to a specific audience yields larger value than streaming it to a random audience. We model the problem using distributed constraint optimization problems (DCOPs), with privacy. Each user has a set of shared variables, and private constraints on those variables. In this context, a local CSP describes a set of interests and constraints on those interests: all participants have stake in the ads being useful, not simply ignored.

1 Introduction

Historically, advertisement providers have used every known trick to get the Internet users' attention: flashing ads, pop-ups, moving windows, multimedia, hogging browser resources. In the most extreme cases, they have resorted to intrusive techniques that go as far as exploiting a vulnerability and installing malware to collect data, or circumventing browser protection (such as pop-up blocking) to show their ads [9]. We address a type of advertising problems, for adds delivered using distributed games. These problems have the properties that (a) participants are motivated to cooperate; (b) they are concerned about privacy; and (c) the constraint defining preferences are naturally distributed.

A distributed constraint optimization problem (DCOP) is the perfect model for such problems. A general DCOP formalism uses: X – the set of variables, D – the domain of those variables, C – the set of constraints, A – the set of agents, I – inputs from those agents defining the constraints and O – the respective outputs for each agent as a function of the solution. Constraints can be either public or private to some of these agents. When agents input the constraint directly, and are informed of the full solution, then the I and O elements are optional, being the identity functions. The case of a single variable is practically a voting problem that can be modeled with DCOPs. More variables require the full DCOP power.

The solution of the DCOP provides incentive for both the clients and the ad providers: clients want their ads to be useful to them, ad providers want their ads to be useful to the client and not simply ignored. If constraints cannot be satisfied, the ad has lower value to the participants - proportional to the number unsatisfied participants and constraints.

The outline of this paper is as follows: Section 2 provides a background. In Section 3, we discuss the bidding process. Section 4 provides discussion and future work and Section 5 concludes.

2 Background

Distributed Constraint Optimization [4] is a general framework to model problems where a group of participants want to find an agreement on an assignment of values to a set of variables, such that the sum of a set of functions (aka weighted constraints) on those variables is maximized. Some of the weighted constraints are private to various participants.

Several versions of this framework are available, with different capabilities for modeling privacy. Among the versions that allow for fine specification of privacy requirements are defined in [3, 6, 2, 1]. We next provide the version in [6], which is used in one the algorithms proposed later and is general enough to encompass the other cases that we study.

Definition 1 (DCOP). *A Distributed Constraint Optimization Problem (DCOP) is defined by six sets (A, X, D, C, I, O) and an algebraic structure F. $A{=}\{A_1, ..., A_n\}$ is a set of agents. X is a set of variables, and D is a set of domains (one for each variable). $I{=}\{I_1, ..., I_n\}$ is a set of secret inputs. I_i is a tuple of α_i secret inputs (defined on F) from the agent A_i. Each input I_i belongs to F^{α_i}. C is a set of weighted constraints. There may exist a public weighted constraint in C, ϕ_0, defined by a function $\phi_0(\epsilon)$ on tuples of assignments ϵ, known to everybody. However, each constraint $\phi_i, i{>}0$, in C is defined as a set of known functions $\phi_i(\epsilon, I)$ over the secret inputs I, and the tuples ϵ of assignments to all the variables in a set of variables X_i, $X_i \subseteq X$. $O{=}\{o_1, ..., o_n\}$ is the set of outputs to the different agents. Let m be the number of variables. $o_i : D_1 \times ... \times D_m \rightarrow F^{\omega_i}$ is a function receiving as parameter a solution and returning ω_i secret outputs (from F) that will be revealed only to the agent A_i.*

A wide range of solvers are available for problems modeled as DCOPs. Some of them are based on constructive distributed search, such as ADOPT [4] and BnB-Adopt [12]. Others are based on dynamic programming, such as DPOP [5]. These algorithms define various trade-offs between efficiency and privacy guarantees. Another family of algorithms strive to achieve maximal privacy at any cost. Among them we mention the Branch&Bound MPC-DisWCSP [8, 6, 7]. These later algorithms can handle the full privacy specifications of the version of the DCOP framework given above, and are used for the experiments described in this paper.

3 DCOP model

We address the application where a set of n users employ an application (such as a networked game) and the provider of the game streams advertisements to a set of m positions in the GUI of the game. The video ads may involve significant bandwidth and therefore we assume that the provider wants to maximize the utilization of his network by multicasting the ads such that all users see the same ads in the same position.

We address first the case of several users negotiating for a stream of advertisements coming from one provider. The provider may sell/resell adds and tries to stream them in such a way as to maximize their impact. Remarkably, the users share the same purpose as the provider, preferring ads that will be useful to themselves. For simplicity, we will first assume that to achieve their purpose, the users act truthfully in declaring their true preferences on ads, whenever needed. However, if possible, they prefer to maintain the privacy of their preferences. The obtained problem fits very well into the mold of a standard DCOP framework, where all the participants collaborate to maximize the sum of the weights of their constraints. It should be noted that when the ad stream is shared among multiple agents, the number of constraints that can be satisfied can be lower.

We model the problem as a DCOP with a set A of n agents (one per user). The provider acts as a broker, that announces a description of the set R of d ads competing for the m slots (positions in the GUI). Those ads are prescreened to match the context of the stream to be viewed. The slots are modeled by m variables in the set $X = x_1, ..., x_m$. Each variable can take values from the set $D = [1..d]$, where each value corresponds to one of the ad descriptions. Each agent holds a set of private weighted constraints on the possible values of subsets of the variables. The accumulated weight given to an assignment by all the constraints of an agent specifies the value/interest that the user would have in viewing the corresponding combination. Such constraints can be specified by the user, or can be computed from the profile of the user and from the description of the ads (generating a high weigh if there is a match and a small weight if there is no match between the profile and the description). There also exists a globally know public constraint, specifying that the n ads should be all different.

The DCOP problem can be solved using any existing solver, based on constructive search or on secure cryptographic protocols. While maximal privacy is achieved by secure multi-party protocols such as MPCDisWCSP4, among the most efficient distributed solvers we mention DPOP. Since DPOP and MPCDisWCSP4 are deterministic protocols, their efficiency can be predicted without experimentation. As such, the efficiency of DPOP for this problem is $2nd^m$ local additions, $2n$ message latencies for messages of size d^m weights. Since one can expect that m, the number of ads simultaneously displayed by the GUI, is small, the computation of DPOP will be reasonable. The efficiency of a secure multi-party computation is experimentally detailed in the experiments section. Algorithms such as ADOPT could be employed to offer a trade-off between the efficiency of DPOP and the security of MPC-DisCSP4 and they should constitute the subject of further work.

In the second case, the provider bids for a single client. The ad stream is private to the client. The ad provider distributes a set of descriptors, one for each ad that may be streamed and displayed in the users' GUI. Each descriptor consist of a set of possible users profiles, and a score. The score estimates how much a user with such a profile will value this ad. A user profile in an ad descriptor can contain regular expressions or a Turing machine that will match a set of possible description. The agent of each user compares the private user profile with the ad descriptors and establishes the value to be associated by the user to each possible configuration of ad displays. The computation occurs in a sandbox [11] and the result of the computation is thrown away.

Below is an example of clients interests and their constraints. The ad provider bids against those clients on the shared stream. The common client interests are A and D. We have performed initial end-to-end experiments with ad optimization for 4 agents. While our initial implementation took 20 seconds, we are confident that improvements can be made.

Client 1 (Age = 21):
interests[a] = 1
interests[b] = 4
interests[d] = 9
	Ad1 constraints:
	$20 \leq$ Age ≤ 25
	interests[a] = 1
	interests[b] = 4
	interests[d] = 7
	occupation := student

Client 2 (Age = 25):
occupation:= engineer
interests[a] = 3
interests[d] = 4
interests[f] = 6
	Ad2 constrains:
	$35 \leq$ Age ≤ 40
	occupation := engineer

Fig. 1. An example of two clients, ad providers and their preferences.

4 Discussion and future work

In the single client model, we ignore cases of multiple conflicting constraints within the same client (for example, a shared family browser), but those limitations can partially be addressed through multiple browser profiles. Temporal constraints are also not discussed. In the multi-client scenario, the worst case is all conflicting interests- therefore, no optimum allocation can be made.

To minimize latency, the ad computations should be done asynchronously so as not to obstruct with the viewing experience. The display of the actual content should be started before the ads are negotiated. Therefore the ads are negotiated and streamed in the background.

Given the malicious history of some advertisers [10], some may still choose to abuse the AdCSP system. In addition, malicious clients can still ignore or skew the bidding process with incorrect preferences.

5 Conclusions

We introduce and evaluate a new application of Distributed Constraint Optimization Problems (DCOPs), namely for modeling and solving the problem of agreeing on the content of a steam of advertisements. In this problem, a set of users of a networked application have agents using their profile to find the preferred combination of ads out of a set announced by the provider(s) of the application. This has to be done while keeping the profiles private to the users. We discuss two alternative ways for modeling the problem. In one alternative, where the variables stand for slots of ads, the solution of the problem specifies directly the solution. In the second alternative the variables model profile items of the users, and the (secure multi-party) solver finds the solution that satisfies the constraints.

An experiment with the second approach shows that the solution can be found with guaranteed privacy withing 20 seconds of distributed computation, using our current implementation of MPC-DisWCSP4 (which is currently based on Shamir secret shares, but could use any other multiparty computation technique, such as homomorphic encryptions for other efficiency/security trade-offs). If the ads are made of long movies, the first 20 seconds of the ads may not be optimized, but any subsequent ads will be obtained as the result of the optimization process, improving both the experience of the users and the efficiency of the advertisement process.

We have shown that DCOPs can be successfully used to improve user experience and ad efficiency, and opened a new problem for the future DCOP research.

References

1. Prashant Doshi, Toshihiro Matsui, Marius Silaghi, Makoto Yokoo, and Markus Zanker. Distributed private constraint optimization. In *IAT*, 2008.
2. Boi Faltings, Thomas Leute, and Adrian Petcu. Privacy guarantees through distributed constraint satisfaction. In *Proceedings of IAT*, pages 350–358, 2008.
3. Rajiv T. Maheswaran, Jonathan P. Pearce, Emma Bowring, Pradeep Varakantham, and Milind Tambe. Privacy loss in distributed constraint reasoning: A quantitative framework for analysis and its applications. *Journal of Autonomous Agents and Multiagent Systems (JAAMAS)*, 2006.
4. P.J. Modi, M. Tambe, W.-M. Shen, and M. Yokoo. An asynchronous complete method for distributed constraint optimization. In *AAMAS*, Melbourne, 2003.
5. A. Petcu, B. Faltings, and D. C. Parkes. M-dpop: Faithful distributed implementation of efficient social choice problems. *JAIR*, 2007.
6. M.-C. Silaghi, A. Abhyankar, M. Zanker, and R. Bartak. Desk-mates (stable matching) with privacy of preferences, and a new distributed CSP framework. In *FLAIRS'05*, 2005.
7. M.-C. Silaghi, B. Faltings, and A. Petcu. Secure combinatorial optimization using DFS-based variable elimination. In *Symposium on AI and Maths*, January 2006.
8. M.-C. Silaghi and D. Mitra. Distributed constraint satisfaction and optimization with privacy enforcement. In *3rd IC on Intelligent Agent Technology*, pages 531–535, 2004.
9. Yi-Min Wang, Doug Beck, Xuxian Jiang, Roussi Roussev, Chad Verbowski, Shuo Chen, and Sam King. Automated web patrol with strider honeymonkeys: Finding web sites that exploit browser vulnerabilities. In *Proc. Network and Distributed System Security (NDSS) Symposium*, 2006.
10. Yi-Min Wang, Ming Ma, Yuan Niu, and Hao Chen. Spam double-funnel: Connecting web spammers with advertisers. In *Proc. International World Wide Web (WWW) Conference*, 2007.
11. B. Yee, D. Sehr, G. Dardyk, B. Chen, R. Muth, T. Ormandy, S. Okasaka, N. Narula, and N. Fullagar. Native client: a sandbox for portable, untrusted, x86 native code. In *Proceedings of IEEE Security and Privacy*, 2009.
12. William Yeoh, Sven Koenig, and Ariel Felner. Idb-adopt : A depth first search dcop algorithm. In *IJCAI DCR Workshop*, 2007.