Multiagent Coordination Using Graph Structured Mathematical Optimization (Extended Abstract)

Arambam James Singh, Akshat Kumar
School of Information Systems
Singapore Management University
arambamjs.2016@phdis.smu.edu.sg,akshatkumar@smu.edu.sg

ABSTRACT

We address the problem of solving mathematical programs defined over a graph where nodes represent agents and edges represent interaction among agents. We focus on the class of graph structured linear and quadratic programs (LPs/QPs) which can model important multiagent coordination frameworks such as distributed constraint optimization (DCOP). For DCOPs, our framework provides a key benefit of modelling functional constraints among agents (e.g. resource, network flow constraints) in a much more tractable fashion. Our framework is also more general than previous work on solving graph-based LPs/QPs as it can model a richer class of objective function and constraints than previous work. Our iterative approach has several desirable properties—it is guaranteed to converge to the optimal solution for LPs, it works for general cyclic graphs, it is memory efficient making it suitable for resource limited agents, and has anytime property. Empirically, our approach provides solid empirical results on several standard benchmark problems when compared against previous approaches.

Keywords

DCOP; Congestion-Games; Graph-Based Optimization

1. GRAPH-BASED OPTIMIZATION

We are given a pairwise graph $G=(V,E)$. The semantics of this graph is same as for DCOPs. Each node $i \in V$ can be considered an agent with an associated random variable x_i. With each edge, there is a potential function θ_{ij} associated, and with each node a unary function θ_i is associated. In addition, our approach provides explicit support for linear constraints defined over nodes and edges of the graph. Specifically, a set of linear constraints M is given. Our goal is to solve the program with the structure in table 1. We next explain the structure of this program.

- Associated with each random variable x_i is a probability distribution $\mu_i(x_i)$. Its contribution to the objective function is $\sum_{x_i \in D_i} [\mu_i(x_i)\theta_i(x_i) + \alpha_i(x_i)\mu_i(x_i)^2]$. In a departure from DCOPs, we allow quadratic terms $\alpha_i(x_i)\mu_i(x_i)^2$ in the objective function, where $\alpha_i(x_i) \geq 0$.

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

$$\max_{\boldsymbol{\mu}} \sum_{i\in V}\sum_{x_i} \mu_i(x_i)\theta_i(x_i) + \sum_{(i,j)\in Q}\sum_{x_i x_j} \mu_i(x_i)\mu_j(x_j)\theta_{ij}(x_i,x_j) + \sum_{(i,j)\in L}\sum_{x_i x_j} \mu_{ij}(x_i,x_j)\theta_{ij}(x_i,x_j) + \sum_{i\in V}\sum_{x_i} \alpha_i(x_i)\mu_i(x_i)^2 \quad (1)$$

$$\text{s.t.} \sum_{i\in \mathrm{V}_m}\sum_{x_i} c_i^m(x_i)\mu_i(x_i) + \sum_{(i,j)\in \mathrm{E}_m}\sum_{x_i,x_j} c_{ij}^m(x_i,x_j)\mu_{ij}(x_i,x_j) = k_m \ \forall m \in M \quad (2)$$

$$0 \leq \boldsymbol{\mu} \leq 1 \quad (3)$$

Table 1: GraphOpt: Graph-based optimization

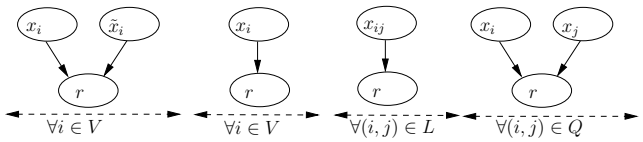

Figure 1: Mixture of BNs for GraphOpt in table 1

Such terms can help model QP problems.

- We are given a partition of the edge set as $E = L \cup Q$. We associate probabilities $\{\mu_{ij}(x_i,x_j) \forall x_i, x_j\}$ with each edge $(i,j) \in L$. The contribution of an edge $(i,j) \in L$ to the objective is $\sum_{x_i,x_j} \mu_{ij}(x_i,x_j)\theta_{ij}(x_i,x_j)$. Intuitively, this forms the *linear* part of the objective in parameters μ_{ij}. The edge set L is known as the set of *LP edges*.
- The contribution of an edge $(i,j) \in Q$ to the objective function is $\sum_{x_i,x_j} \mu_i(x_i)\mu_j(x_j)\theta_{ij}(x_i,x_j)$. This forms the quadratic part of the objective function; and the edge set Q is known as the set of *QP edges*.

The overall objective function defined over the edges E is shown in program (1).The distributions $\boldsymbol{\mu} = \{\mu_i, \mu_{ij}\}$ are the parameters to optimize by solving the program (1). We next show how solving program (1) can be recast as a likelihood maximization problem using decision making-as-inference strategy used previously [3, 2].

We create a mixture of simple Bayesian networks (BNs) corresponding to the GraphOpt problem in table 1. Fig. 1 shows the structure of each of four types of BNs. For first BN type, we introduce a *binary* reward variable r whose conditional probability is directly proportional to $\alpha_i(x_i)$ providing a link between the likelihood in this model and our objective

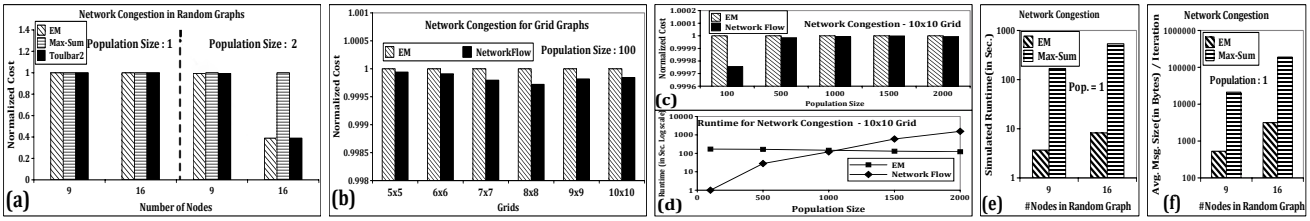

Figure 2: Solution quality, Simulated Runtime and Message size comparisons for congestion-based routing instances

in (1). The parameters of this model are set as:

$$P(r=1|x_i,\tilde{x}_i)=\begin{cases}\frac{\alpha_i(x_i)+1}{K} & \text{if } x_i=\tilde{x}_i\\ \frac{1}{K} & \text{otherwise}\end{cases} \quad (4)$$

where K is a large enough positive constant such that each $\frac{\alpha_i(x_i)+1}{K}$ is less than 1 to make it a probability. For the 2nd, 3rd and 4th type of BN components, we have:

$$P(r=1)=\sum_{x_i}\mu_i(x_i)\hat{\theta}_{x_i}, P(r=1)=\sum_{x_i,x_j}\mu_{ij}(x_i,x_j)\hat{\theta}_{x_ix_j}$$

$$\text{and } P(r=1)=\sum_{x_i,x_j}\mu_i(x_i)\mu_j(x_j)\hat{\theta}_{x_ix_j} \quad (5)$$

THEOREM 1. *Maximizing the likelihood $P(r=1;\boldsymbol{\mu})$ of observing the variable $r=1$ in the mixture model of figure 1 subject to constraints M in (2) is equivalent to solving the* GRAPHOPT *problem.*

The above result shows how the principle of likelihood maximization (LM) can be used to solve our program (1). The well known Expectation Maximization (EM) algorithm is a general approach for maximum likelihood parameter estimation. Therefore, we use it to maximizes the log-likelihood of our parameters $\boldsymbol{\mu}$. In each iteration, the EM algorithm solves the following convex optimization problem:

$$\min_{\boldsymbol{\mu}^\star}\sum_{i\in V,x_i}-\log\mu_i^\star(x_i)\Big[\hat{\theta}_{x_i}\mu_i(x_i)+\frac{2\alpha_i(x_i)\mu_i(x_i)^2+2\mu_i(x_i)}{K}+\sum_{j\in \mathrm{Nb}_Q^i}\sum_{x_j}\hat{\theta}_{x_ix_j}\mu_i(x_i)\mu_j(x_j)\Big]-\sum_{(i,j)\in L,x_i,x_j}\hat{\theta}_{x_ix_j}\mu_{ij}(x_i,x_j)\log\mu_{ij}^\star(x_i,x_j) \quad (6)$$

$$\text{s.t.}\sum_{i\in \mathrm{V}_m}\sum_{x_i}c_i^m(x_i)\mu_i^\star(x_i)+\sum_{(i,j)\in \mathrm{E}_m,x_i,x_j}c_{ij}^m(x_i,x_j)\mu_{ij}^\star(x_i,x_j)=k_m\ \forall m$$

where $\boldsymbol{\mu}$ denote parameters from previous iteration, $\boldsymbol{\mu}^\star$ are parameters to optimize. The above optimization problem can be solved using a message-passing scheme over the underlying graph, resulting in a distributed approach.

2. EXPERIMENTAL RESULTS

We test our approach on instances based on congestion-based routing problem and standard DCOP benchmarks based on random graphs and sensor networks. We compare our EM based solver with several standard approximate DCOP solvers such as Max-Sum (MS), DSA, and an efficient centralized solver Toulbar2 [1] which provides a strong baseline for solution quality comparisons. We set iterations for each approach (EM, MS, DSA) to 1000. Each data point is an average over 10 instances. We always show normalized solution quality. For the Toulbar2, we set 30 min. limit.

Congestion aware routing: We tested on random and grid shaped graphs with varying agent population size. We compare DCOP solvers against EM. For DCOP solvers, we first convert the problem instance to a DCOP instances and the constraints tables in those instances are exponentially sized requiring $O(N^{2d})$ space, where d is degree of a node in the graph and population size is N. Due to this reason, DCOP solvers did not scale up to solve large networks. Fig. 2(a) shows comparison on small 9 and 16 node graphs with small population size. All the approaches, except DSA, provided good solution for $N=1$. EM always achieved optimal solution, same as Toulbar2. DSA failed for these problems and did not find any feasible solution satisfying the flow constraints highlighting how the presence of functional hard constraints adversely affects the accuracy of previous approximate solvers. On larger grid graphs and higher population size, none of the DCOP solvers scaled up. For larger grid graphs in fig. 2(b), we provide comparisons against an optimal network flow based solver [5] that can solve symmetric congestion games with linear congestion cost. The cost of EM's solution is only marginally worse than the network flow solver showing the accuracy of the QP approximation.

For larger population N, results in fig. 2(c,d) show that the optimal solver scales poorly with the increasing N. The runtime of the network flow solver increases exponentially with the increasing N, whereas EM has nearly constant (centralized) runtime ($\approx$180 sec) with varying N. These set of results confirm that for problems with functional constraints, it is more tractable to solve a graph-based math program that handles functional constraints explicitly. Fig. 2(e) show the simulated runtime for small 9 and 16 node instances solvable using DCOP solvers. This result shows that EM is orders of magnitude faster than MS. As EM's message-passing structure and MS's message-passing are not equivalent, for fairness sake, we show total *average* network load per iteration of MS and per outer loop of EM in fig. 2(f). That is, for EM, per iteration load counts messages exchanged in a single outer loop and *all* the inner loops. Fig. 2(f) clearly shows that EM has significantly lower network overhead.

DCOP instances: We also tested EM on DCOP benchmarks including random grid graphs and sensor network problems [4]. The grid size varied from 5×5 till 10×10. For these graphs, the domain size was 4. Our results on these problems show that our approach is competitive with existing DCOP solvers. However, in the presence of network flow-based functional constraints such as for congestion-based routing, our approach provided significants speedups over DCOP solvers.

Acknowledgments: This work is supported by the research center at the School of Information Systems at the Singapore Management University.

REFERENCES

[1] D. Allouche, S. de Givry, and T. Schiex. Toulbar2, an open source exact cost function network solver. Technical report, INRA, 2010.

[2] S. Ghosh, A. Kumar, and P. Varakantham. Probabilistic inference based message-passing for resource constrained DCOPs. In *International Joint Conference on Artificial Intelligence*, pages 411–417, 2015.

[3] A. Kumar and S. Zilberstein. MAP estimation for graphical models by likelihood maximization. In *Advances in Neural Information Processing Systems*, pages 1180–1188, 2010.

[4] R. Maheswaran, M. Tambe, E. Bowring, J. Pearce, and P. Varakantham. Taking DCOP to the real world: Efficient complete solutions for distributed event scheduling. In *International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 310–317, 2004.

[5] C. Meyers and A. Schulz. The complexity of welfare maximization in congestion games. *Networks*, 59:252–260, 2012.