Using the Max-Sum Algorithm for Supply Chain Formation in Dynamic Multi-Unit Environments

(Extended Abstract)

Michael Winsper and Maria Chli
Aston University
Aston Triangle
Birmingham, United Kingdom B4 7ET
{winsperm, m.chli}@aston.ac.uk

ABSTRACT

The max-sum loopy belief propagation (LBP) algorithm was shown in [4] to produce strong results in a simple decentralised supply chain formation (SCF) scenario where goods are traded in single units. In this paper, we demonstrate the performance of LBP in a multi-unit SCF scenario with additional constraints. We also provide experimental analysis of LBP's performance in dynamic scenarios where the properties and composition of participants are altered while the algorithm is running. Our results suggest that LBP continues to produce strong solutions in multi-unit scenarios, and that performance remains solid in a dynamic setting.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Economics, Algorithms, Experimentation

Keywords

Supply Chain Formation, Loopy Belief Propagation

1. INTRODUCTION

Computational approaches to SCF model potential supply chain participants as individual computational agents which express their capabilities and costs through a mechanism which determines the subset of agents capable of forming the most efficient supply chain. Although centralised SCF techniques [1] have allowed for multi-unit exchanges for some time, the existing state of the art in decentralised [4, 3] SCF only model simple scenarios where goods are exchanged in single units. Additionally, [4] does not model the effect of changes to the properties or composition of participants once the process has begun. In this paper, we propose a framework for the representation of multi-unit supply chains and extend the LBP-based technique for decentralised SCF presented in [4] to the multi-unit case. We also present a set of experiments analysing the performance of LBP in a dynamic environment, where changes to the properties and composition of participants occur while the algorithm is running.

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), 4-8 June 2012, Valencia, Spain.

Our results demonstrate that LBP is capable of producing efficient allocations over a range of network topologies in both static and dynamic environments.

2. MODEL

The use of task dependency networks (TDNs) for the representation of supply chains was originally proposed in [3]. For the first time, we extend this TDN representation to the multi-unit case by modelling input to output good ratios, production capacities and consumer desired good quantities. An example of the extended representation is shown in Figure 1. Values below producers and consumers represent reserve prices and production capacities, and consumption values and desired consumable good quantities. Edges from goods to producers are labelled with the producer's input to output ratio for that good. A producer with a single input and an input ratio of 2 for that good requires two units of that good in order to produce one unit of its output.

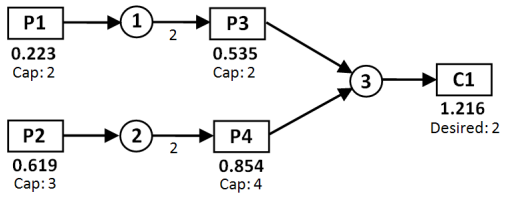

Figure 1: The Simple supply chain TDN from [3] extended to the multi-unit case. Producers (P[x]) and a consumer (C1) are represented by rectangles, while goods are represented by circles. Edges indicate potential flows of goods.

Producers and Consumers At initialisation, each producer is assigned a production capacity which specifies the maximum number of units each producer is able to produce of its output good, and an input to output ratio for each of their inputs. In order to produce one unit of their output good, producers are required to acquire a number of units of each of their input goods equal to their ratio for that good. A producer cannot produce its output good unless it acquires the necessary quantities of all of its input goods. Producers attach a reserve price R_p to their output good, which is linear with the number of units of its output good that it produces. Consumers seek to acquire a number of units of their consumable good no greater than their desired consumable good quantity. In each network, each consumer is assigned a static consumption value V_c representing the

valuation the consumer holds for obtaining a single unit of its consumable good. The total value a consumer receives is linear with the number of goods it obtains.

3. APPLYING THE MAX-SUM ALGORITHM

The max-sum algorithm is a variant of loopy belief propagation (LBP), a decentralised and distributed approximate inference scheme involving applying Pearl's belief propagation algorithm [2] to graphs containing cycles. It uses iterative stages of message passing as a means for estimating the maximum a posteriori assignment; in our case, this corresponds to the network-wide state configuration that maximises Eq. 3. Each state encodes a combination of purchases and sales which may be made by an agent. States are associated with costs - the *unary cost*, representing the cost to the allocation of the agent being in that state, and the *pairwise cost*, which encodes the compatibility of two states of neighbouring agents. At each iteration of the algorithm, every node in the graph sends a message to each of its neighbours, representing the sender's beliefs about the potential cost to the total efficiency of the network of each of the recipient's states. This is calculated using Eq. 1, where x_v is a state of recipient j, $bel_i(x_u)$ is sender i's belief in its own state x_u, $m_{j\to i}(x_u)$ is the message passed from j to i in the previous step about i's state x_u, and $g_{ij}(x_u, x_v)$ is the pairwise cost of states x_u and x_v. Once values have been calculated for all of j's states, i passes the message to j.

$$m_{i\to j}(x_v) = min_{x_u}\Big(bel_i(x_u) - m_{j\to i}(x_u) + g_{ij}(x_u, x_v)\Big) \quad (1)$$

Once all nodes have sent a message to each of their neighbours, nodes then update their beliefs about their own states based upon the content of the messages they received using Eq. 2, where $f_i(x_u)$ is the unary cost of i's state x_u, and $m_{j\to i}(x_u)$ are the messages received from i's set of neighbours N_u about state x_u in the previous step. The process of message passing and belief update continues until the beliefs of each node stabilise. For more information on applying LBP to SCF, we refer the reader to [4].

$$bel_i(x_u) = f_i(x_u) + \sum_{j\in N_u} m_{j\to i}(x_u) \quad (2)$$

Allocation Before allocation is performed, each agent determines their *final state* - the state, when beliefs stabilise, which the agent believes holds the lowest cost. Once the final states of each of the agents have been determined, we classify producers which successfully sell their output good and consumers which acquire their consumable good as *active*. We calculate allocation values using Eq. 3, where C is the set of active consumers C, A_c is the number of goods acquired by consumer c, P is the set of active producers and M_p is the number of goods manufactured by producer p.

$$Val = \sum_{c\in C} V_c A_c - \sum_{p\in P} R_p M_p \quad (3)$$

4. RESULTS

We perform two sets of experiments, examining the performance of LBP in both static and dynamic environments. In the static environment, we compare LBP with a multi-unit implementation of the SAMP-SB auction protocol from [3], extended by using multiple copies of each agent to represent capacities and desired good quantities. This representation does not allow for the use of input to output good ratios, so for fair comparison we also test LBP with all ratios set to 1, referred to as ratioless LBP. We test each technique over 100 runs on each of the networks from [3]. We vary input ratios (drawn from $[1 \dots 2]$), consumer desired goods (from $[2 \dots 3]$), reserve prices (from $U(0, 1)$), and production capacities ($[4 \dots 5]$) between each run, discarding runs in which the optimal allocation value, determined using mixed integer programming, is non-positive. Consumption values are fixed at the per-network values given in [3] over every run. In the dynamic environment, at a number (drawn from $[6 \dots 10]$) of randomly chosen steps during each run we randomly change one of the aforementioned properties of a single agent, introduce a new producer or consumer, or remove a producer. We present our results in terms of average efficiency, calculated by dividing the total allocation values produced by each method over 100 runs on each network by the maximum available value over the same 100 runs.

Table 1: Average efficiency in each network produced by LBP and the SAMP-SB auction protocol from [3] on the networks from [3]. A result of 1.000 equates to the capture of 100% of available efficiency.

Network	Static LBP *ratioless*	Static LBP *with ratios*	Dynamic LBP *with ratios*	Static SAMP-SB *ratioless*
Simple	1.000	1.000	0.911	0.999
Unbalanced	0.962	0.872	0.713	0.964
Two-Cons	0.986	0.983	0.801	0.963
Bigger	0.969	0.813	0.520	0.995
Many-Cons	1.000	1.000	0.989	0.425
Greedy-Bad	0.91	0.839	0.793	0.666

Table 1 shows the average efficiency produced by each method. We see that both static LBP-based methods are able to match or outperform SAMP-SB on the majority of networks. As expected, LBP finds the optimal allocation 100% of the time in static scenarios on acyclic networks, while still being able to produce highly efficient allocations on more loopy networks. We also see that LBP tended to perform better when input to output good ratios are not present; this is to be expected since the presence of ratios serves to constrain the number of solutions available. In the dynamic setting, we see that for most networks, average efficiency is roughly comparable to the results produced in a static environment.

5. REFERENCES

[1] J. Cerquides, U. Endriss, A. Giovannucci, and J. Rodríguez-Aguilar. Bidding Languages and Winner Determination for Mixed Multi-Unit Combinatorial Auctions. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 469–476, 2007.

[2] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, volume 1. Morgan Kaufmann, 1st edition, 1988.

[3] W. Walsh and M. Wellman. Decentralized Supply Chain Formation: A Market Protocol and Competitive Equilibrium Analysis. *Journal of Artificial Intelligence Research*, 19:513–567, 2003.

[4] M. Winsper and M. Chli. Decentralized Supply Chain Formation using Max-Sum Loopy Belief Propagation. *Computational Intelligence*, In Press, 2012.