Decentralised Supply Chain Formation: A Belief Propagation-based Approach

Michael Winsper and **Maria Chli** [1]

Abstract. Decentralised supply chain formation involves determining the set of producers within a network able to supply goods to one or more consumers at the lowest cost. This problem is frequently tackled using auctions and negotiations. In this paper we show how it can be cast as an optimisation of a pairwise cost function. Optimising this class of functions is NP-hard but good approximations to the global minimum can be obtained using Loopy Belief Propagation (LBP). Here we detail a LBP-based approach to the supply chain formation problem, involving decentralised message-passing between potential participants. Our approach is evaluated against a well-known double-auction method and an optimal centralised technique, showing several improvements: it obtains better solutions for most networks that admit a competitive equilibrium[2] while also solving problems where no competitive equilibrium exists, for which the double-auction method frequently produces inefficient solutions.

1 INTRODUCTION

Agent-based computational approaches to supply chain formation model potential supply chain participants as rational self-interested computational agents. These agents deliberate between themselves, typically either through negotiations [5, 2] or auctions[3, 4], about the subset of agents capable of forming the most efficient supply chain. In this paper, we propose a loopy belief propagation-based (LBP) approach to decentralised supply chain formation which is capable of producing efficient results over a range of network topologies from [3]. Using LBP, we are able to produce results comparable to that of a centralised approach whilst working in an entirely decentralised manner. Finally, the use of message passing allows us to take full advantage of the graphical structure of our networks.

In section 2, we provide details of our model, inspired by work previously conducted by [3], and explain how we use the LBP algorithm to determine allocations in our model. Section 3 describes our experiments, while in section 4 we show our results and compare them to the results obtained by [3] and the optimal value.

2 MODEL

Following [3], we model our supply chain networks as bipartite directed acyclic graphs. There are two types of node: producers and consumers - represented by rectangles in our network diagrams (e.g. Figure 1) - and goods, represented by circles. Edges between participants signify the ability for those entities to produce or consume goods. An edge from a participant to a good indicates that the participant is capable of producing the good, while an edge leading from a good to a participant means that the participant is able to consume the good. Goods represent a single unit of an indivisible commodity.

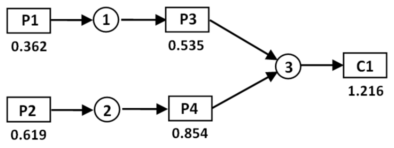

Figure 1. A sample supply chain network, from [3]. Producers (P1,P2,P3,P4) and consumers (C1) are represented by rectangles, with goods represented by circles. Edges between vertices show possible flows of goods. Numbers below producers indicate production costs, while numbers below consumers indicate consumption values.

Agents Our supply chain networks are made up of multiple producer agents aiming to supply a good to one or more consumer agents. Producers are capable of producing a single unit of a single type of output good, and to do so are required to have obtained a single unit of each of their input goods. In producing their output good, producers incur a production cost. Consumers require a single unit of a single good from their set of consumable goods. In each network, each consumer is assigned a static consumption value V_c, representing the valuation the consumer places upon obtaining one of its consumable goods.

States Due to the fixed structure of the networks, for each agent there exist a finite number of purchases and sales in which the agent is viable. We encode each of these tuples of exchanges as states, with each state defining a list of suppliers and a buyer for producers, and a single supplier for consumers. For example, a possible state for producer P3 in Figure 1 is “Buy from P1 and sell to C1”. The number of states an agent possesses increases with the number of participants able to supply/ consume its input/output good(s). We also allow for the *inactive state*, where the agent does not acquire or sell any goods.

2.1 Cost Function

We allow for two distinct types of cost in our model: unary costs, and pairwise costs. Our method minimises the function given below:

$$\epsilon(x_1, \ldots, x_N) = \sum_{v \in V} f_v(x_v) + \sum_{(u,v)\in E} g_{uv}(x_u, x_v) \qquad (1)$$

Where $\epsilon(x_1, \ldots, x_N)$ is the set of agents, $f_v(x_v)$ is the unary cost of agent v being in state x_v, and $g(u, v)\ (x_u, x_v)$ is the pairwise cost of linked agents u and v, being labeled with states x_u and v_v. With all else equal, the lower the cost, the more efficient the allocation.

[1] Aston University, Birmingham, United Kingdom, email: {winsperm,m.chli}@aston.ac.uk

[2] Competitive equilibrium as defined in [3] is used as a means of classifying results on certain networks to allow for minor inefficiencies in their auction protocol and agent bidding strategies.

Unary Cost Each agent associates each a cost with each of its states. For all agents, the cost of being in the inactive state is zero. For producers, the cost of active states is equal to their production cost. Consumers assign a cost $0 - V_c$ to active states, where V_c is the consumer's consumption value.

Pairwise Cost To calculate the pairwise cost for two states, we assess their compatibility. Two states are compatible if both are inactive states, *or* the lists of sellers and buyers align such that neither is trying to buy or sell the same good, *or* the states align so that agent u wants to sell to agent v, v's list of sellers includes u, and neither is inactive, and vice versa. If the states are compatible, the pairwise cost is zero; if they are incompatible, the pairwise cost is infinite.

2.2 Loopy Belief Propagation

LBP [1] involves the iterative passing of messages by nodes to each of their neighbours encoding which state the sender believes the recipient should be in. Once all agents have passed a message to each of their neighbours, each agent updates its beliefs based upon the content of the messages it received. This process continues until the beliefs of the agents become stable, at which point we determine the final state of each agent and perform allocation.

Belief Update For each of agent u's possible states, we use equation 2 to calculate u's belief in that state. At initialisation, each agent holds a belief of zero about each of its states.

$$bel_u(x_u) = f_u(x_u) + \sum_{w \in N_u} m_{w \to u}(x_u) \tag{2}$$

$bel_u(x_u)$ denotes agent u's belief in its state x_u, $f_u(x_u)$ is the unary cost of u being in state x_u, and $m_{w \to u}(x_u)$ are the messages received from u's set of neighbours w about state x_u.

Messages At each step, each agent in the network passes a message to each of its neighbours, consisting of a vector of values representing the sender's beliefs about each of the recipient's states.

$$m_{u \to v}(x_v) = min_{x_u}(bel_u(x_u) - m_{v \to u}(x_u) + g_{uv}(x_u, x_v)) \tag{3}$$

Equation 3 shows the process of calculating a message to be passed from agent u to agent v. $bel_u(x_u)$ corresponds to u's belief in its state x_u, while $m_{v \to u}(x_u)$ is the message passed from v to u about state x_u in the previous round of messages and $g_{uv}(x_u, x_v)$ is the pairwise cost of agents u and v being in states x_u and x_v. We repeat this process for each of u's states, and use the minimum of these values in the message. This process is then repeated for each of v's states.

Final States and Allocation Before we can perform allocation, we assign a "final state" to each agent – the state which, at convergence, the agent believes holds the lowest cost. Once the final states of each of the agents have been determined, we measure the value of the allocation by the equation given below, where C is the set of consumers in the allocation, V_c is the consumption value of consumer c, P is the set of producers in the allocation, and PC_p is the production cost of producer p. This is equivalent to equation 1.

$$\sum_{c \in C} V_c - \sum_{p \in P} PC_p \tag{4}$$

3 EXPERIMENTS

We test our method over a variety of network structures, taken from [3]. Upon initialisation of each of the networks, the production cost of each producer is drawn from the interval $U(0, 1)$. These values are changed after each run, while consumption values remain static. As in [3], we gather 100 results for each network, discarding runs with a non-positive optimal result. We compare the value of our allocations to the optimal value, calculated using local search, and to the results of the auction protocol given in [3], categorising our results into efficiency categories as follows: *Negative*, where the production costs of active producers exceeds the value that the consumer(s) obtain from acquiring their consumable good; *Zero*, where our algorithm does not reach convergence; *Suboptimal*, where a positive non-optimal solution was found; and *Optimal*, where LBP achieved the same efficiency as the centralised benchmark. These efficiency classes are identical to those in [3].

4 RESULTS

Table 1. Distribution of efficiency classes from LBP and SAMP-SB. Classes are Negative, Zero, Suboptimal and Optimal.

	LBP % of instances				SAMP-SB % of instances			
Network	Neg	Zero	Sub	Opt	Neg	Zero	Sub	Opt
Simple	0.0	0.0	0.0	100.0	0.0	0.3	0.0	99.7
Unbalanced	8.0	1.0	0.0	91.0				
CE					5.0	1.0	7.0	87.0
No CE					100.0	0.0	0.0	0.0
Two-Cons	0.0	0.0	0.0	100.0				
CE					11.0	0.0	6.0	83.0
No CE					18.0	0.0	78.0	4.0
Bigger	0.0	0.0	0.0	100.0	0.0	0.0	4.0	96.0
Many-Cons	0.0	0.0	0.0	100.0	27.0	0.0	56.0	17.0
Greedy-Bad	0.0	7.0	0.0	93.0				
CE					4.0	0.0	21.0	75.0
No CE					100.0	0.0	0.0	0.0

We see from Table 1 that our approach is able to improve upon SAMP-SB's performance over all networks tested. Due to the absence of producer surplus in our model, we make no attempt to distinguish between the existence of competitive equilibrium (CE) or otherwise in our results. However, even if we compare our results with the best case for SAMP-SB, using only those results where CE exists, we are still able to show a clear advantage in the proportions of our runs showing optimal efficiency, with marked reductions in negative and suboptimal allocations.

REFERENCES

[1] B.J. Frey and D.J.C. MacKay, 'A revolution: Belief propagation in graphs with cycles', in *Neural Information Processing Systems*, pp. 479–485. MIT Press, (1997).

[2] T. Jiang, R. Foley, X. Yao, and H. Tianfield, 'An extended contract net mechanism for dynamic supply chain formation and its application in china petroleum supply chain management', *Multiagent and Grid Systems*, **2**(2), 183–207, (2006).

[3] W.E. Walsh and M.P. Wellman, 'Decentralized supply chain formation: A market protocol and competitive equilibrium analysis', *JAIR*, **19**, 513–567, (2003).

[4] W.E. Walsh, M.P. Wellman, and F. Ygge, 'Combinatorial auctions for supply chain formation', in *Proc. of the 2nd ACM conf. on Electronic Commerce*, pp. 260–269, (2000).

[5] M. Wang, H. Wang, D. Vogel, K. Kumar, and Chiu D.K.W, 'Agent-based negotiation and decision making for dynamic supply chain formation', *Eng. Appl. Artif. Intell.*, **22**(7), 1046–1055, (2009).