Competitive Sourcing for Internet Commerce

Steven P. Ketchpel
Computer Science Department
Stanford University
Stanford, CA 94305

Hector Garcia-Molina
Computer Science Department
Stanford University
Stanford, CA 94305

Abstract[1]

In electronic commerce on the Internet, a customer can choose among several competitive suppliers, but because of the nature of the Internet, the reliability and trustworthiness of suppliers may vary significantly. The customer's goal is to maximize its utility, by minimizing the expense required to fulfill its request, and maximizing its probability of success by some deadline. To this end, the customer creates a request strategy, *describing which suppliers to contact under what conditions. In this paper we describe models for representing request strategies complete with supplier reliabilities, delivery timeliness profiles, and customer deadlines. We also develop decision procedures for selecting request strategies that maximize expected utility under certain scenarios, and more efficient heuristics that approximate the optimal solution.*

1 Introduction

When a customer purchases goods over a network such as the Internet, he or she may have many possible sources for the desired goods. These sources may charge different prices and have different reliabilities, and the customer must select one (or more) sources based on this information. In some cases, the purchases are time-critical, so that if the goods are not received before a deadline, they are worthless. In other cases, the customer is trying to purchase a bundle of goods and only is satisfied if all of the individual goods (also called *conjuncts*) are obtained.

For example, a U.S. news network is trying to organize its response to a fast-breaking story in Europe. It needs to send key personnel to the scene, hire local labor for the behind the scenes work, and rent satellite up-link time. The response is time-critical–a slow reaction means that rival stations draw huge audiences. Yet managing costs is also important, and reliability and immediate response from suppliers come at a premium price. How should the network choose among various suppliers, and when should they place redundant orders to increase reliability?

This paper offers guidance for the customer faced with such a competitive sourcing problem. The customer creates a *request strategy*, a plan of which suppliers to contact and when they should be contacted. For example, the news station might have the request strategy of first contacting its local European affiliate for a camera crew, but plan to exercise the more expensive option of hiring a local movie company if the crew still has not arrived an hour before air time. Examples for the rest of the paper will be drawn from the domain of information goods, where a customer seeks the answer to a query which may require multiple sources, and may have redundant suppliers of some portions of the query.

The representation of request strategies and expected utility calculations are the contributions of this paper. A decision theoretic framework is used to select an optimal request strategy, one that maximizes the customer's expected utility.

In economics, this problem of selecting appropriate suppliers is known as "sourcing". The decision theoretic approach has been applied in the economics, operations research, and industrial engineering literature to sourcing problems, as described in Section 2. However, the properties of e-commerce cause some differences with the previous work in these fields. Among the most important differences are:

- The "make-to-order" environment of mass customization. The results for each request need to be interactively generated as a result of particular parameters not known until the customer's order arrives. The model of maintaining an inventory of finished goods to supply a known demand is simply not appropriate. Therefore, issues of real-time performance and the ability to meet a deadline established by the customer become much more important.

- Lower overhead for becoming a supplier or for splitting orders. Customers can have many suppliers offer bids for a contract, or split a "bundled" order across multiple suppliers more easily, resulting in more parties per transaction. However, by contacting suppliers that the customer has not had personal experience with, supplier reliability becomes much more problematic, especially in a networked environment where participants can be anonymous.

As the number of suppliers and goods increases, the number of possible strategies grows very quickly. The introduction of time and deadlines contributes to a combinatorial explosion. So, in some cases, the search procedures are very computationally expensive, and it is not realistic to assume that an optimal solution may be found. Even in these cases, however, a customer can compare the expected utility of two proposed solutions, and determine if a new proposal is an improvement. Similarly, if business needs demand that the request strategy take a particular form (e.g., all orders placed at the same time) the reduction in dimensionality may make the it possible to efficiently optimize for one parameter. In building the framework for multi-good sourcing problems, the upper bounds on the complexity of strategies show how much more complex they are than the single-good problems. Within this framework, one can now search for optimizations or explore heuristics that offer efficient approximations of the optimal solution. Some initial investigations in this area appear in Section 7.

The goal of this paper is to provide algorithms for determining optimal ordering strategies, under the following conditions:

1. Each supplier may charge a different price and have a unique reliability.

2. Customers may require conjunctions of goods and receive no benefit for a subset of the goods.

3. Customers may face hard deadlines, obtaining no benefit if the goods are not received in time.

4. Suppliers may require customers to pay at the time the order is placed. The customer loses his money if he pre-pays for an order from an unreliable supplier.

To accommodate these conditions, an expressive representation for ordering strategies is required. In particular, the presence of deadlines requires a specification of *when* goods should be ordered. The presence of conjunctive requests requires knowing *under what conditions* (e.g., what other goods have been received or have not yet been received) an order should be placed.

2 Related Work

A broader view shows this problem to be an instance of the more general problem of "sourcing", that is, finding suppliers that can provide the inputs for the production of a good. In the field of economics, this falls under the area of industrial organization. Competing schools argue for either *sole sourcing* where only a single supplier is considered, or *competitive sourcing* where multiple suppliers may be asked to fulfill a particular order. Proponents of sole sourcing (e.g., [2]) argue that by developing close ties with one supplier, coordination costs are reduced, quality improves, and supplier performance is maximized. On the other hand, proponents of competitive sourcing (e.g., [7]) claim that relying on a single supplier puts a firm at the mercy of that supplier, and to reduce risk, a firm should maintain relationships with multiple suppliers, playing them off each other to obtain the lowest price and best service. Space constraints prevent an exhaustive list of related work (see [4]), but we describe a few key works.

Leidy [5] is the first to consider unreliable suppliers. He considers several cases, where the reliability and cost of suppliers are the parameters under investigation. He defines strategies for problem instances with one reliable supplier, one unreliable supplier, n equally unreliable, equal price suppliers, and one high priced reliable supplier, with lower price unreliable suppliers.

Leidy's main focus is ensuring that the customer obtains an adequate quantity of the input good (always a single good). His result is to recommend the appropriate amount the buyer should order from each supplier, pointing out that by spreading the order across multiple (independent) suppliers, the risk may be reduced. Leidy's goal is to determine the optimal ordering strategy, which reduces to the quantity to order from each supplier. Leidy does not consider the case of multiple inputs, nor payments that must be made at the time the order is placed. Deadlines are not mentioned (though could be captured in terms of reliability estimates.)

Gurnani, Akella, and Lehoczky [3] consider the multi-good case where there is a particular type of multiple suppliers per good: in addition to one supplier for each of the goods, there is another supplier, the "joint" supplier, that provides all of the needed goods. Chu, Proth, and Xie [1] considers multi-good problems with multiple suppliers. However, their model still differs from ours along several dimensions. In particular, in their model:

1. Inventory and backlogging costs are charged.
2. Only single suppliers are considered for each good.
3. The assembler is not given the option of simply ignoring an order which may be too hard (costly) to fulfill.
4. The optimal time points for all components are determined at the start, and cannot depend upon the actual arrival or non-arrival of other components in the request process.

3 Sourcing Strategies

Our decision theoretic framework permits several different instantiations, depending on features such as whether multiple suppliers are allowed for each input, and so on. This section describes the different possibilities and assumptions that underlie them.

The first consideration is whether the buyer faces a deadline. The simpler cases assume there is no deadline: The buyer can make a request from one supplier, wait until either the input arrives or the supplier is deemed unreliable, and then order from a second supplier, and so on. In contrast, a buyer may face a strict deadline, which may require requests to be made in parallel rather than serial fashion.

The second consideration is the number of conjuncts the buyer is attempting to acquire. There is a special case if that number is only one, as certain risks (obtaining a subset of the needed goods) are eliminated. Therefore, the relevant options are one conjunct or more than one conjunct.

The third consideration is the number of suppliers for each conjunct. If there are never multiple suppliers for a particular conjunct, the situation is somewhat simplified. As in the previous consideration, all of the complexity of handling n suppliers is introduced with the second supplier. A particular supplier is denoted (i, j) where i indicates the input the supplier will supply, and j is the ranking of this supplier among all of the suppliers of i.

A longer version of this paper [4] adds the further consideration of whether payment is due at the time of order (in which case a payment to an unreliable supplier is a loss) or at delivery.

3.1 Assumptions

The use of a decision theoretic framework requires the buyers to have a substantial amount of information. For example, the buyers must know the utility gain U of fulfilling a goal (and get 0 if the goal is not fulfilled). The buyers also know the price that the suppliers charge (denoted $C_{i,j}$, which is denominated in utility units) and know (or have estimates of) suppliers' performance in *delivery profiles*. Formally, a delivery profile, $Sat_{i,j}(t)$, is a function over natural values of t for the j-th supplier of input i.[2] $Sat_{i,j}(t)$ is the buyer's estimate of the probability that the request will be satisfied (the input will arrive at the buyer) at or before t time units after making the request. The delivery profile also measures a supplier's *reliability*, denoted $R_{i,j}$, which corresponds to $Sat_{i,j}(\infty)$. Delivery profiles may be constructed based on a customer's prior experiences with a supplier, or, if this is an unknown supplier, on average statistics, information from a consumer reporting agency, or the supplier's self-reported delivery date.

Further assumptions include the buyers having sufficient resources to pay the costs associated with making the requests, and documents being of adequate "quality" to fulfill the customer request. If there is a chance that it is not, that can be captured as part of the delivery profile. Delivery time is assumed to be independent between suppliers.[3] If a single entity is a supplier for two different inputs, it is assumed the two inputs may be treated separately, and delivery performance will be independent between them. (In this case, the single supplier will have multiple aliases.) A final assumption is that requests will be answered in the order received by the source, and orders are conveyed reliably to the source. Therefore, making multiple requests from the same source for the same document is not beneficial.

The following section describes the single input case with one supplier (with and without a deadline). Section 5 adds additional suppliers to the one input case. Section 6 addresses the problem of multiple inputs. Section 7 considers easier computations to approximate the optimal strategy, and Section 8 offers conclusions.

[2] The use of discrete values instead of continuous permits the exhaustive enumeration of strategies. A continuous formulation would also be possible, though the optimization problem would possibly be more complex.

[3] Although this assumption is questionable (especially if the suppliers are two brokers obtaining the document from the same source), it is commonly made in the sourcing literature.

4 One input, one supplier

With only one input and one supplier, the buyer's question is very simple: "Do I make the request right away, or not bother at all?" There are no alternative requests, and there is no benefit to waiting.

4.1 No deadline for customer

In the simplest case, covered by [5], payment is due only upon delivery. Therefore, there is no risk in placing the request, and it is made right away (assuming the utility gain exceeds the price charged by the supplier). Only when payment is due upon order does the supplier's reliability becomes an issue. The customer must decide if the chance that the supplier will be unreliable (and merely pocket the money) outweighs the benefit of a successful transaction. The customer's request strategy is to place the order right away if its expected utility from the request is positive:

$$(U * R_{1,1}) - C_{1,1} > 0$$

4.2 Customer has deadline

A customer may face a deadline D beyond which the customer receives no utility. If payment is due on order, the situation is similar to the previous case, except the customer has to consider not only those suppliers that are unreliable ($R < 1$), but also those that might be too slow to meet the deadline $Sat(D) < 1$. In this case, the request strategy is to place the order if

$$(U * Sat_{1,1}(D)) - C_{1,1} > 0,$$

and to not place the order otherwise.

5 One input, multiple suppliers

With multiple suppliers, the buyer faces a more interesting choice. The question is no longer "Should I make the request of the supplier?", it becomes "Which (if any) suppliers should I make the request of?"

5.1 No buyer deadline

Without a deadline, the buyer is not concerned with time, so the suppliers should be approached serially. The order in which the suppliers is approached is critical: The customer wishes to get the cheapest available supplier, but the cost should include the loss due to unreliable suppliers. The customer should produce a ranking of intended order of approaching the suppliers of input 1: $(1,1)$, $(1,2)$, and so on. The order is based on the expected utility gain of approaching $(1,j)$ (same as single supplier case):

$$(U * R_{1,j}) - C_{1,j}$$

The customer should approach first the supplier providing the highest gain, wait to see if that supplier is reliable, and if not, approach the next on the list, and so on. The customer should stop approaching suppliers when either:

1. The document is provided.
2. The expected utility gain is nonpositive.

Up to this point, it is relatively clear that the strategies proposed are optimal. A formal proof would postulate a model for all possible strategies, and then argue that the ones presented here beat all others. For instance, in the one input, multiple supplier, no deadline case of this subsection, strategies that use a different order of suppliers or that do not wait for the previously contacted supplier to deliver or "time out" would do worse than the strategy advocated here. However, the sections that follow do present formal models for representing all possible strategies, since instead of easily finding the best strategy, an exhaustive search is used to find the optimal one. These request strategy models could be used to argue optimaility in the simpler scenarios.

5.2 Buyer deadline

In this case, the customer is weighs the same two considerations as in the single supplier case: the suppliers may be unreliable or simply too slow. The presence of multiple suppliers (even for a single document) offers the customer the chance to make redundant requests, which might be worthwhile for a valuable document when the delivery time is uncertain. The customer may achieve savings by waiting to see whether one of the cheap, fast suppliers succeeds before spending money on a more reliable, but more expensive supplier. The method proposed here for finding a request strategy is exponential in the number of suppliers and the time limit until the deadline. However, it is effective, in that given enough computational power, an optimal solution will be found.

Recall that, by assumption, a later request made from one supplier will never be filled before an earlier request of that same supplier. Therefore, a customer will never make more than one request from a supplier. A request strategy, then, is an assignment of a request time (from a discrete set of time points) to each supplier. If it is not rational to make a request of a particular supplier, then the time of request will be set to some point beyond the customer's deadline.

A new function called $RT(i,j)$ defines the *request time*, or time point at which supplier (i,j) is contacted. (In this "single input" section, the first subscript will always be 1.) At time t, if $RT(i,j) < t$, then supplier (i, j) was contacted at the indicated time. If $RT(i, j) = t$, then the request strategy recommends making the request in the current time step. If $t < RT(i, j) < D$, where D is the customer's deadline, the request strategy expects to make the request at that future time point (although the goal may be achieved before that point is reached). Finally, if $RT(i, j) > D$, no request is expected for this supplier.

5.2.1 Request strategies

The buyer's request strategy governs the buyer's actions for each time step. In order to compute the optimal request strategy, it is necessary to project forward in time to forecast the likely outcomes of actions suggested by this strategy. Based on these forecasts, the expected utility for a request strategy can be calculated. The number of distinct strategies is $|S|^D$, where S is the set of suppliers, as each supplier may be assigned a time point between 1 and D, inclusive. (A request slated for time point D will never be made; it is assigned only to "unhelpful" suppliers.) The expected utility gain for a particular strategy is:

$$U * (1 - \prod_{(1,j) \in S, RT(1,j) < D} (1 - Sat_{1,j}(D - RT(1,j)))) - \sum_{(1,j) \in S, RT(1,j) < D} (C_{1,j} * \prod_{(1,k) \in S} (1 - Sat_{1,k}(RT(1,j) - RT(1,k))))$$

The first term of this expression reflects the expected utility gained from obtaining the document. It is a combination of the utility upon success, U, and the probability of success. A number of independent requests are being made, and the success of any one is sufficient to achieve the goal. Therefore, the probability that none of the requests succeeds before the deadline is simply the product of the probabilities that each has failed to respond in the amount of time allotted to it (the deadline minus the time the request was made). The second term reflects the cost for each request that the buyer makes. Note that cost of making a request is incurred only if that request is made (i.e., it happens before the deadline and none of the requests before it has succeeded in obtaining the goods at the time the request is due to be made). The probability of some other request successfully acquiring the goods before this request is made is based on the delivery profile given the time difference between the scheduled times for the requests.

Finding an optimal strategy can be accomplished by searching through the space of all strategies. A valid value for RT (between the current time and the deadline, or a value at the deadline, indicating no request is made) is assigned for each supplier, resulting in a complete request strategy. These request strategies are subjected to the utility analysis from the formula above, and the request strategy with the highest value is selected. In Section 7 we discuss approximations which reduce the size of this search space.

6 Multiple Inputs, Multiple suppliers

If there is more than one document, then acquiring a single document provides no utility unless the rest of the conjuncts can also be acquired. Therefore, the calculation must take into account the expected cost to acquire all of the conjuncts, as well as the chance that the acquisition will fail due to one or more conjuncts being unavailable. The set of desired goods will be labeled G_1 through G_k.

6.1 No buyer deadline

With multiple inputs, the customer must decide not only which supplier to contact, but which input to try to acquire first. Without a deadline, there is no advantage gained by making requests in parallel, and doing so may lead to unnecessary costs if one of the conjuncts is unobtainable. Therefore, the optimal request strategy is necessarily a serial ordering of requests to suppliers. The customer can consider in advance the available options for the first request (placing orders with any of the suppliers), followed by the two possible outcomes (either the document arrives or it does not), followed by the second stage requests for either of those outcomes (any of the uncontacted suppliers), the possible outcomes there, and so on, until all of the conjuncts have been acquired, or all of the suppliers for one conjunct have failed to deliver. The paths through this tree correspond to a series of requests and their outcomes. By starting at the leaves and working back up the tree, the value of each interior node may be found (as described below). These values guide the request strategy of the customer. Constructing the optimal strategy requires the consideration of all the different alternatives.

Figure 1 shows a partial tree for two conjuncts (a and b) and two suppliers for b, called $(b, 1)$ and $(b, 2)$. Nodes below the dashed node are omitted, but would be symmetric to those shown. The process described for this example generalizes to any number of conjuncts and suppliers. At the root of the tree (not shown in Fig. 1) is the customer's decision of which (if any) of the conjuncts to request first.

After the customer makes a request (represented by circle nodes on the graph in Figure 1) "nature" (actually, the previously requested supplier) makes a move

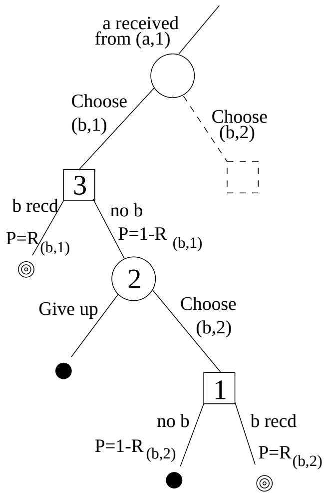

Figure 1: A fragment from a decision tree for a two conjunct, two-supplier per conjunct problem. The subtree below the "dashed" node has been omitted for clarity.

(nodes represented by squares in Figure 1). Nature's choice is between sending the document or not sending it, and the customer estimates those selections will be taken with probabilities corresponding to the supplier's reliability, indicated with labels on the relevant edges in Fig. 1. After learning the outcome of that request, the customer faces another choice. If the request was fulfilled, the customer can select one of the other conjuncts to obtain. In the case of two conjuncts, this translates into a forced choice of requesting the missing document from one of its suppliers. If the request failed, the customer can choose to stop, to try another supplier for that conjunct, or select a different conjunct.

The utility of each of the outcomes (leaf nodes) is known. It is simply the positive goal utility if the full conjunction is obtained (nodes terminating with a "bulls-eye" in Fig. 1), minus the sum of costs for all of the conjuncts requested. If the goal is not achieved (the filled-in circle leaves), the cost is still incurred, but with no positive utility gain to offset the costs.

In order to determine the request strategy the customer should execute, the value of the tree is calculated in a bottom up fashion. At the square nodes where "nature" (the suppliers) controls the outcome (decides whether the document is delivered or not), the value of node is computed from the combination of its children, weighted according to their probabilities. For example in Figure 1, the node labeled "1" is reached after the customer receives a; goes on to request b from $(b,1)$, and when that one fails, from $(b,2)$ as well. Node 1 corresponds to whether the request to $(b,2)$ is fulfilled or not. The customer estimates its probability of being filled as $R_{b,2}$. If the request is fulfilled, the customer has a positive utility of U for fulfilling the conjunction, but has paid for requests from $(a,1)$, $(b,1)$, and $(b,2)$. If the request is not fulfilled, the resulting utility to the customer is negative–the price charged by $(a,1)$ plus that of $(b,1)$ and $(b,2)$. So, assuming payment is due on order, the expected utility at the node labeled "1" is

$$(U * R_{b,2}) - C_{a,1} - C_{b,1} - C_{b,2}$$

At the node labeled "2" in Fig. 1, the customer has obtained conjunct a, and knows that $(b,1)$ has failed to deliver b. It is deciding whether to contact $(b,2)$ or quit. The utility of quitting is the total cost incurred so far, $-(C_{a,1}+C_{b,1})$. Given that at Node 2, the customer makes the choice among the options, the customer will choose the child with the higher value. This value is given to Node 2 as well, that is $\max((U * R_{b,2}) - C_{a,1} - C_{b,1} - C_{b,2}, -(C_{a,1}+C_{b,1}))$. This value for Node 2 captures all of the different possibilities below it in the tree. Its value is now backed up to Node 3, and is combined probabilistically with the other child of Node 3, where $(b,1)$ fulfills the request, weighted according to $R_{b,1}$.

This process of backing values up the tree continues, weighted by the reliability probabilities at the suppliers' nodes, or selecting the maximum at the customer's nodes. The values backed up to the root instruct which request (if any) the customer should make first. Upon learning the result of that request, the appropriate branch is followed, and the option yielding the maximal expected utility from that node is selected, with the process repeating until a terminal node is reached. The size of the tree is exponential in the number of suppliers, as we show in [4].

6.2 Buyer has deadline

The combination of a deadline with conjunctive goals makes decisions much harder. As in the previous case, a request for one document may be contingent on the arrival of a second. However, it now may be necessary to make requests in parallel, and base decisions on inferences about the expected arrival or non-arrival of documents as a function of time. As before, it is assumed that a later request of the same supplier will never be answered before an earlier one, so it is never rational to make multiple requests of a single supplier. Moreover, if a single supplier offers two different desired conjuncts, the requests may be handled independently, as though they were from two different suppliers.

The tree representation of Section 6.1 is insufficient in an environment with deadlines, because it does not permit parallel requests. Therefore, a different representation for request strategies based on *triggers* is used. A trigger is a rule that watches for certain conditions at certain times. If the condition is met, the trigger fires, and the supplier associated with the trigger is contacted. The language used for the trigger conditions is a conjunction of various documents. So, for example, the supplier $(b, 1)$ may be contacted at time $t = 5$ if document a has been received. The restriction persists that each supplier is contacted at most once, so once a trigger is activated, it should suppress all later triggers for the same supplier (though other suppliers of the same conjunct may be contacted). Note that triggers are weaker in one respect than the tree representation: the strategy cannot distinguish which one of multiple suppliers ended up providing the document, only whether some has or none has. This restriction seems reasonable in that it does not matter which supplier ultimately provides the document, since they are all equivalent documents. For example, a customer desiring documents a and b may make a request for b from supplier $(b, 1)$ only if a has been received, but not distinguish whether a came from $(a, 1)$ or $(a, 2)$ in making that decision.

Formally, a request strategy Δ is a set of triggers $T_1, T_2, \ldots, T_k$ for each supplier. A trigger T_i is a pair (t, DV), where t is the time point at which the trigger is applied, and DV is the *document vector* guarding the trigger. A document vector is a binary vector of length $|G|$ (the cardinality of the goal conjunction), where a 1 in position DV_k indicates that document G_k has been received, and a 0 in that position indicates it has not been received. If the guard condition is satisfied at the time point of the trigger, the trigger is activated, and the document is requested from the supplier. For example, trigger T_3 below contacts supplier $(b, 1)$ at time 5 if a is held and b is not.

Identifier	Supplier	Time	a	b
T_3	$(b, 1)$	5	1	0

As before, a simple but expensive exhaustive search through the space of strategies finds the optimal strategy. The strategy space is large, but each member can be enumerated. For each supplier, there are at most $2^{|G|}$ triggers (each of the different combinations of 0 and 1 for length $|G|$). Each of these triggers can be either present or absent at each of the time points up until the deadline D, for a total of $D^{2^{2^{|G|}}}$ strategies per supplier. With $|S|$ suppliers, the total number of strategies is $|S|^{D^{2^{2^{|G|}}}}$. This (admittedly impractical) naive estimate is an overestimate of the relevant strategies, because it introduces many triggers which can never be activated. For example, a strategy may be based on having a particular conjunct, yet no triggers for that conjunct exist prior to this trigger.

With each of these different request strategies, one can calculate its expected utility (described below), choose the one which offers the highest utility, and execute it (make requests based on which triggers fire at that time step). At the next time step, the appropriate triggers are again considered, with the customer's holdings updated by the conjuncts that arrived in the previous time step. The process repeats until the goal is achieved or the deadline is reached.

The expected utility gain for request strategy Δ:

$$U * P(G \text{ achieved by } D) - \sum_{(i,j)\in S} (C_{i,j} * P(Req(i, j)))$$

A slight variant of notation in Section 5.2, $P(Req(i, j))$ is the probability that the request strategy contacts (i, j). Unfortunately, this utility calculation relies on two terms that are not directly available: the probability that the goal is achieved by the deadline and the probability that a request is made of a supplier. In order to calculate those probabilities, we construct a Bayes net [6] representing the probabilistic relationships among the variables. The Bayes net is a compact representation of a family of conditional independence statements, allowing the efficient calculation of the probability that a random variable under consideration takes on a given value, with other variables' values observed as evidence.

The variables that we are concerned with (represented as nodes in the Bayes net) are: a distinguished *goal node*, a *status node* for each goal conjunct at each time point until the deadline (denoted below by conjunct@time), and a *trigger node* for each trigger. All of the nodes are Boolean, representing whether the goal is achieved or not, whether a document is held or not, and whether a trigger fires or not. The net represents the progression of time, having a separate "layer" of status nodes for each time point.

The major concepts of the Bayes net representation of a request strategy are described in the context of an example. The general principles for constructing the Bayes net are summarized at the conclusion of the example.

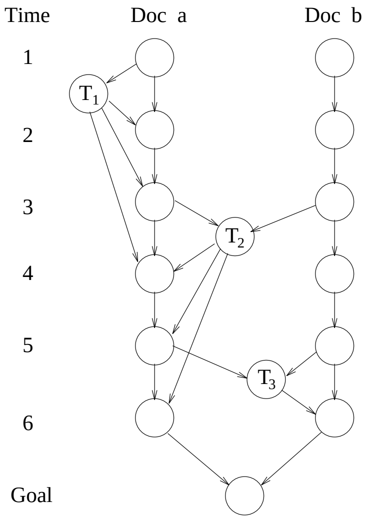

Figure 2: Bayes net representation of strategy of example in Section 6.2.1

6.2.1 Example

A customer is trying to obtain 2 different documents, called a and b before a deadline of time 6. There are two suppliers for a, called $(a,1)$ and $(a,2)$. The single b supplier is called $(b,1)$. The delivery profiles are, for the sake of simplicity, identical for $(a,1)$ and $(a,2)$, with a delivery equally likely to arrive in each of the 3 time steps following the request, though the reliability is only 66%. Formally, the *Sat* profile is (.22, .44, .66, 0*). For $(b,1)$, the response is in the first time step if at all, (.5, 0*).

The customer is considering the utility of a strategy shown in Table 1. This strategy places requests with $(a,1)$ (Trigger T_1) immediately. If $(a,1)$ has not responded by time 3, then it places a request with $(a,2)$ (Trigger T_2). If either a supplier has responded by time 5, the strategy tries supplier $(b,1)$ (Trigger T_3).

Table 1: Request strategy for example in Section 6.2.1

Identifier	Supplier	Time	a	b
T_1	$(a,1)$	1	0	0
T_2	$(a,2)$	3	0	0
T_3	$(b,1)$	5	1	0

The completed Bayes net structure appears in Figure 2. Its construction process is described here. Each of the two major vertical "chains" is composed of 6 status nodes that tell whether the customer has a conjunct at a time point. For example, the node in the upper left corner describes the status of document a at time 1. The node can have one of two values, `true` (also denoted $a@1$) and `false` ($\overline{a@1}$). A probability is assigned according to the likelihood that $a@1$ takes on either of these values. Since we are starting without a, the `false` value is assigned probability 1.

Links connecting nodes in the Bayes net correspond to causal relations. Each (non-root) node has a *conditional probability table (CPT)* which assigns probabilities to the node's values based on the values of its parents. For example, the links connecting a status node to the subsequent status node reflect that whether the document is held at one time step has a strong effect on whether it is held in the next time step. In this example strategy, the status node $b@2$ depends only on $b@1$ (since no order is placed for b at time 1), so if the customer has b at time 1 ($b@1$ holds), it also has it at time 2. Inversely, if it is not held at 1, it will not be available at 2. This relationship is captured in the CPT of $b@2$:

	$b@1$	$\overline{b@1}$
$b@2$	1	0
$\overline{b@2}$	0	1

In addition to the status nodes, there is a goal node, which indicates whether the goal is satisfied by the deadline. It is at the center of the bottom of Figure 2. Its value (either `true`, the goal was met, or `false`, it was not) is determined by the status nodes of the two conjuncts at the deadline, time 6. The goal is met only if both of the conjuncts have been obtained (have `true` values). Therefore, the goal node is a pure `AND` node:

	$a@6b@6$	$a@6\overline{b@6}$	$\overline{a@6}b@6$	$\overline{a@6b@6}$
G	1	0	0	0
$\overline{G}$	0	1	1	1

The third type of node, in addition to status nodes and the goal, is a trigger node. The value of trigger node (`true` if it fires, `false` otherwise) is determined by the values of its relevant document statuses at the time the trigger is applied. The first trigger for any supplier is straightforward. It is activated whenever the value of its parents (relevant documents) take on the necessary values. For example, T_3 fires (is `true`) when a is held and b is not. The corresponding CPT for T_3 is:

	$a@5b@5$	$a@5\overline{b@5}$	$\overline{a@5}b@5$	$\overline{a@5b@5}$
T_3	0	1	0	0
$\overline{T_3}$	1	0	1	1

The second and subsequent triggers for a supplier have (in addition to the relevant documents) all of the earlier triggers for that supplier as parents, which prevent the new trigger from firing if any of the older triggers have, to ensure each supplier is contacted at most once.

The output values of the triggers affect the status nodes of the documents they request. For example, if T_1 is activated at time 1, then according to $(a, 1)$'s delivery profile, a has a 22% chance of arriving at time 2. Therefore, $a@2$ is affected by both T_1 and $a@1$. The full CPT for $a@2$ is:

	$a@1T_1$	$a@1\overline{T_1}$	$\overline{a@1}T_1$	$\overline{a@1T_1}$
$a@2$	1	1	.22	0
$\overline{a@2}$	0	0	.78	1

The first two columns indicate that if a is held at time 1, it will still be held at time 2. The last column indicates that if it was not held and T_1 does not fire, it cannot be held at time 2. The third column updates the chance that a is held by the probability (from the delivery profile) that it arrives 1 time unit after being requested.

When two separate suppliers have been contacted, their delivery profiles must be combined to generate the resulting status node. For an example, see [4]. There is an added complexity when more than one trigger is used for a single supplier at one time step (to implement disjunctive triggers). That supplier's contribution must not be double counted.

The generic rules for connections among the nodes are as follows:

1. Each of the goal conjunct@deadline status nodes is connected to the goal node.

2. Each of the goal conjunct@t status nodes is connected to the corresponding goal conjunct@$t+1$ status node.

3. Each of the relevant documents in the trigger's document vector is connected to the trigger. The connection is from the document's status node at the trigger time point to the trigger node, and does not depend whether the document is required (has a 1 value) or forbidden (0 value).

4. Trigger nodes are connected to all subsequent trigger nodes for the same supplier (to provide suppression if an earlier trigger fires).

5. Trigger nodes are connected to subsequent status nodes for the requested document, for as many time steps as the delivery profile indicates there is a positive probability of the requested document arriving.

The generic rules for the CPTs are as follows:

1. *Goal node:* The CPT for the goal node is a pure `AND` node. The value of the goal is `true` iff all its parents are `true`.

2. *Trigger node:* The CPT for the trigger node is also a pure `AND` node. The value of the trigger is `true` iff all of its "1" value document nodes are `true` and all of its "0" value documents are `false`, and all of the prior triggers for that supplier are `false`.

3. *Status node:* The CPT depends on the status of the document in the previous time step (if it was `true` at $t-1$, it is `true` at t.) It also depends on the possible deliveries from suppliers that may have been contacted recently. For each supplier, if any of the triggers that contact that supplier fired, the probability of the status node is increased by the incremental *Sat* entry (which must be calculated from the cumulative values in the profile) for the time step corresponding to the amount of time that has elapsed since the request was made.

With the completed Bayes net, it is possible to calculate the expected utility of a strategy. The algorithms described in [6] permit the calculation of any variable's probability given the prior probabilities of root nodes and instantiated values of "evidence" variables. The desired calculation is

$$U * P(G \text{ achieved by } D) - \sum_{i,j \in S} (C_{i,j} * P(Req(i,j)))$$

The term $P(G \text{ achieved by } D)$ is directly available as the probability of the goal node in the Bayes net. The $P(Req(i,j))$ for a given supplier can be calculated by summing the probability of all the different conditions under which a trigger for that supplier fired. This set is exponential in the size of the number of triggers, but the probability of each can be calculated, giving the total probability that the supplier is contacted before the deadline. The insertion of these quantities into the formula above permits the calculation of the utility for any request strategy.

This approach also works if there is only a single provider per input, with no modification.

7 Approximations

In the multi-good case without deadlines, a decision tree is generated, with a new supplier being selected from among all the suppliers that have not yet been contacted. There are two ways that the size of this set (and consequently, the branching factor of the tree) may be controlled. One approach is to cluster all of the requests for one conjunct together. For example, if the customer were trying to acquire a and b, a request strategy of $(a, 1)$, $(b, 1)$, $(a, 2)$ would never be considered, because the two requests for conjunct a are separated by one for b. The intuition behind this restriction is that in order to fulfill the request all of the conjuncts must be obtained, so once the process is started for one, attempts to acquire it should continue until it is obtained or the cost is so high that giving up on the whole conjunction is the rational choice.

The effectiveness of this "Within Conjunct" heuristic has been evaluated through simulation. The resulting performance compared well to the optimal strategy. In a random distribution with relatively small variance among the conjunct costs, the heuristic approach failed to find the optimal solution in less than 1 in 1000 instances. The full results are reported in [4].

A second heuristic can be combined with the "Within Conjunct" to further reduce the number of strategies considered. This second approach, which we refer to as the "Rank" heuristic, is to totally order the sequence in which the suppliers of a given conjunct will be contacted. One possible ranking heuristic is the expected cost of obtaining the input from that source alone (or an infinite number of independent sources with identical characteristics). So, for example, if a source is 80% reliable and charges 10 for the document, then the expected cost of that supplier would be $10 + .2 * 10 + .2^2 * 10 + .2^3 * 10 + \ldots$, or the sum of the infinite geometric series, $C_{i,j} * 1/R_{i,j}$. For a particular conjunct the customer knows which supplier should be contacted, the one with the lowest expected cost among those not yet contacted. Overall, the rank heuristic works well, although performance degrades as the number of suppliers increases relative to the number of conjuncts. Details may be found in [4].

For dealing with deadlines, one approach is simply to avoid search all together, and directly assign the times at which requests should be made based upon heuristic guidelines. Two rules for assigning these request times in single good cases have been investigated. The simpler heuristic requires that all requests be made at time 0. No incremental information is used, all decisions are made before the acquisition process begins. This heuristic results in lost utility (compared to optimal) amounting to about 25.5% of the sum of the costs of the suppliers for problems with 3 conjuncts and 4 suppliers. A second, more sophisticated heuristic, uses more information from the delivery profile to choose the request time. It strictly dominates the first heuristic, and results in a penalty of about 3.5% of the sum of the suppliers' costs. As the size of the problem instance grows (along both number of suppliers and number of conjunct dimensions), the performance of both heuristics decreases linearly. Details may be found in [4].

8 Conclusion

This paper has presented a model for making sourcing decisions, for scenarios such as information commerce. These decisions may depend on the number of inputs, number of suppliers and their reliability, and whether the customer has a deadline or not. Different request strategy representations (e.g., decision trees, Bayes nets) are appropriate depending on the characteristics of the problem instance. For each representation, a method for evaluating the expected utility of a request strategy is given. This capability is the basis of selecting a request strategy. A customer could decide to "trade up" to an alternative request strategy that has a higher expected value. Similarly, in a parameterized strategy, the customer may be able to find the optimal value of the parameter by evaluating the utility of strategies with its legal values.

References

[1] Chengbin Chu, Jean-Marie Proth, and Xiolan Xie. Supply management in assembly systems. *Naval Research Logistics*, 40(7):933 – 949, December 1993.

[2] W.E. Deming. *Out of the Crisis*. MIT Center for Advanced Engineering, 1986.

[3] Haresh Gurnani, Ram Akella, and John Lehoczky. Optimal order policies in assembly systems with random demand and random supplier delivery. *IIE Transactions*, 28(11):865–878, November 1996.

[4] Steven P. Ketchpel. The networked information economy: Applied and theoretical frameworks for electronic commerce. Ph.D. Thesis, Stanford University, 1998.

[5] Michael P. Leidy. Hedging across suppliers under stochastic input deliveries. *Economic Notes*, 21(1):39–59, 1992.

[6] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[7] M. Porter. *Competitive Advantage*. Free Press, 1985.