Broadcasting on a Budget in the Multi-Service Communication Model

Gene Itkis
NDS Technologies Israel
Jerusalem, Israel
itkis@ndsisrael.com

Ilan Newman
Math and Computer Science
Haifa University, Haifa, Israel
ilan@mathcs.haifa.ac.il

Assaf Schuster
Dept. of Computer Science
Technion, Haifa, Israel
assaf@cs.technion.ac.il

Abstract

In this paper we introduce the MULTI_SERVICE model of network communication. This model attempts to capture recent communication technology trends, such as aspects of quality-of-service and their relation to the emerging technology of automatic pricing, e.g. for Internet services. The MULTI_SERVICE model differs from related models by taking communication and the activation time of a service provider into account, thus restricting parallelism to better fit reality. Hence, our model extends and refines previous successful models for network communication.

We consider the application of this model to communication problems, where the service providers are point-to-point message passing agents each with its characteristic pricing policies and speed. We give some insights and an algorithm for optimal dissemination of information in this model when given a fixed, limited budget.

1 Introduction

1.1 Background

Many information services are available on the Internet with the touch of a fingertip. Typically, different service-providers are characterized by their corresponding costs and guaranteed quality of service. A client seeking for services (typically in an online/ repeatedly emerging needs) faces the need for making an intelligent choice of service providers. The variety of choices and the related complexity involved in taking such decisions in this multi-parameter environment call for the development of computational models and efficient algorithms for these purposes.

In the future, the services that will be provided over the Internet are likely to include two types parameters:

Price is what the service provider charges for its service.

Quality-of-service is a certain quality-related guarantee (speed, reliability, etc.) that the vendor is committed to provide for the service at the corresponding price.

For example, consider courier vs. regular mail where the corresponding quality-of-service is the speed of delivery.

Optimization becomes more complicated when the same task is to be performed repeatedly several times, and there is a given total budget which is not likely to cover the best available service for all of these uses ("to which of the addresses should I use the courier?"). Choice may get even more complicated when the optimization of the global objective function depend in some non-trivial way on the order of the chosen service providers, and their corresponding performance (as in the case of broadcasting on a budget, considered below).

Commonly, a shopper will be faced with a set of k service providers characterized by a set of pairs of numbers (c_i, q_i), $i = 1, ..., k$, where c_i corresponds to the cost of using the ith provider, and q_i corresponds to the quality of service of the ith provider. This model may be used to deal with the general shopping question, involving the coordination of several different tasks towards a common goal, and finding service providers for each of these tasks.

Given a certain task and an allocated budget, the user will have to choose a subset of the service providers and a schedule for their activation which will guarantee that the task will be performed while meeting the budget and quality of service constrains. For example, the work of Etzioni et. al. [10] may be viewed in this framework, where the quality-of-service corresponds to the probability of success and the cost corresponds to the needed time. Alternatively, given a set of performance constraints (e.g., deadlines) the user needs to select and schedule the service providers to minimize the cost.

In this work we consider *communication providers*, of which the service quality is commonly measured by *speed*. The typical task is how to disseminate information to many cooperating clients as fast as possible using a variety of point-to-point communication providers and a limited budget (e.g using the media of telephony with the different providers being the various long distance companies/cell phone/cable/internet/ fax etc.). Each provider is specified by its cost (which in principal may depend on various parameters of the point-to-point message that is to be transmitted), and has a guaranteed time it takes to complete each transmission. In addition, an extra parameter, which is crucial to this mode of operation, is a time overhead or 'setup time' incurred in order to activate a provider (e.g the time it takes to contact the provider headquarters, the time it takes from a 'mouse click' until the homepage of the provider arrives etc.). This parameter limits the ability of the client to activate in parallel many providers. The mode of operation is the following: At a certain step, the client may chose to activate a certain service provider. This will take an overhead of l steps, after which the client may immediately chose to activate another provider for a different message, while the previous provider might be still performing the previous communication.

We concentrate hereafter on the problem of broadcasting a single message, hence this overhead, denoted by l is taken to be fixed for all the transmissions throughout the paper.

1.2 Broadcasting on a budget

The problem of *broadcasting on a budget using multiple communication media*, or simply, *broadcasting* is the following: A message M stored initially at a certain processor P_0, is to be communicated to n other processors. There is a set $S = \{(c_i, q_i) || i = 1, \ldots, k\}$ of k point-to-point communication providers, where c_i is the cost of using the ith service provider, and q_i denotes the time in which the ith provider guarantees to deliver message M to an indicated (single) destination. Calling a communication provider involves an overhead (setup time) of l steps. While sending another processor the message M, the sending processor may ask the receiving one to be in charge of broadcasting M to a subset of the remaining destinations. Hence the broadcast schedule has a tree-like structure, as shown in Figure 1. Each schedule is associated with the time it takes to complete the broadcast (the time until the last of the n processors receive the message), and the cost which is the sum the costs of each of the transmissions. In one variant of the optimization problem a total budget B is given, and a schedule is looked for, so that its time will be the shortest while still meeting the budget B. Another variant is to broadcast when some bound, or a deadline, is put on the time to complete it, and the objective is to minimize the cost. As it turns out these two problems can be

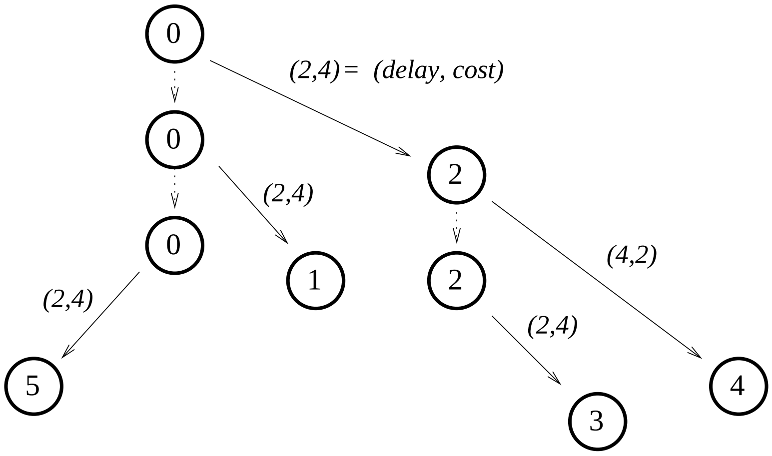

Figure 1: A tree representing a broadcast schedule for $n = 5$ processors. The algorithm for scheduling uses two service providers of costs $2, 4$ and respective delays $4, 2$. The arrows between tree nodes of different id represent sending a message, and the vertical arrows represent a setup time of l steps between successive message transmissions by the same processor. Initially, processor 0 transmits the message to processor 2 in step 1 using a provider of delay 2 and cost 4, then after a setup time of l steps it uses the same service to send the message to processor 1. Meanwhile processor 2 uses the provider of delay 4 and cost 2 to transmit the message to processor 4, etc. The total cost of this algorithm is 18 and its total time is $\max\{2l + 2, l + 4, 6\}$.

solved by using the same tools.

The rest of the paper is as follows. In Subsec. 1.3 we list some related work. In the Sec. 2 we give formal definitions (Subsec. 2.1) and general properties of broadcast schemes in our model (Subsec. 2.2). Section 3 gives algorithms for computing efficient broadcasting schedules. Subsection 3.1 considers the case of just two service providers, while Subsection 3.2 solves the general problem for arbitrary number of service providers. The complexity issues of these algorithms are addressed in Subsec. 3.3.

1.3 Related Work

Our communication-oriented service model was inspired by, and is generalizing, the successful POSTAL communication model that was introduced by Bar-Noy and Kipnis in [3]. Indeed the two models coincide when the budget is unlimited, or when there is only a single provider. Querying each service provider at a time corresponds to the SINGLE_PORT communication model [4, 3, 2, 5]. Using multiple service providers has some similarity with the MULTI_PORT communication model [1, 6, 7].

Using multiple communication providers is reminiscent of the MULTI_MEDIA communication model [9]. None of the previous communication models, however, is dealing with the quality-of-service dimension.

The analysis of Lopsided Trees was given in [8]. The relation of these trees and broadcast algorithms is discussed in [9]. Algorithms that were given in these works are used here as building blocks.

A previous work that does take quality-of-service into account consider information gathering and searching on the Internet [10]. Similar to this work, [10] is also dealing with speed related quality-of-service in the context of data gathering. In [10] the authors consider a set of service providers each with its corresponding cost and probability of providing the service successfully. Two models are presented: the *cost model*, where the cost of a search is the sum of the costs of the queried sources, and the *reward model* in which there is a certain delay for each source and the price is the reward given to the first source which supplies the answer. However, the MULTI_SERVICE model is drastically different from these two models in the use of parallelism: In the models of [10] there is unbounded parallelism - namely the client can activate as many service providers it wants. The task is then finished when the first of service providers successfully ends. Solution is thus a linear schedule of the service providers. In the MULTI_SERVICE model, at each point, a node (the client or a commissioned processor) may activate only one service provider. The parallelism is obtained by the tree-like structure of the schedule - namely, the fact that commissioned service providers that already have been activated get to activate other service providers for the same original task.

2 Model

2.1 Notations and Definitions

Let n denote the number of processors which must receive the broadcast message. Since each processor needs to receive it exactly once, exactly n messages must be sent during the broadcast. Let the set of available service providers be given by the set of pairs, $S = \{(c_i, q_i) \| i = 1, \ldots, k\}$, where the i-th provider's charges c_i for a point-to-point message transmission and guarantees a quality of the service of q_i denoting (an upper bound on) the transmission time.

W.l.o.g., we assume that $c_i > c_{i+1}$ and $q_{i+1} > q_i$, for $1 \leq i \leq k-1$, i.e. the service providers are listed in the order of decreasing costs and the cheaper service providers are slower (a provider which is more expensive and slower than another available one would not be used and can be ignored). Also, we assume that the delay, q_i for every provider, includes the setup for the next message, so

$q_i > l$ for all i. Since the leaves do not need to setup for the next message, the broadcasts' times we compute below can be reduced by l steps. We ignore this saving for the sake of simplicity.

A *broadcast schedule* indicates where, in what order, and using which provider should each processor forward the message M when it receives it. We represent it as a tree (e.g., as in Figure 1): The nodes represent processors at different times, and each edge represents either message transmission or setup (cf. [9]).

More formally: A broadcast *schedule* T is a directed tree whose edges are labeled by providers (either indexed, or may be labeled by the provider cost and time of transmission), or labeled as setup edges. Each node v in the tree corresponds to some processor p_v at a time t_v when p_v is receiving and/or sending a message. The root r corresponds to processor p_0 (the original sender of the message) at time $t_r = 0$. Each non-leaf node v has exactly two out-going edges:

- A *transmission* edge $e = \vec{vw}$ labeled by the, say ith provider, corresponds to a message sent by processor p_v at time t_v to $p_w(\neq p_v)$, and received at $t_w = t_v + q_i$.
- A *setup* edge $e = \vec{vu}$ is always of weight l and corresponds to some processor $p_v = p_u$ during time from t_v to $t_u = t_v + l$ when p_v is preparing to send the next message.

In the figures we represent the setup edges with the dotted vertical lines, while the solid lines depict the transmission edges.

In our model we restrict each node to have at most one in-going edge, at most one setup edge and at most one transmission edge.[1] For a broadcast T, its *cost* $B(T)$ is the sum over all transmission edges, of the costs associated with the corresponding providers. $S(T)$ is the set of service providers used by T. $\|T\|$ denotes the number of receiving processors. The time of T, denoted by $height(T)$ is the $max_{v\in T}\ t_v$.

Define $H(n,B,S) = \min\{height(T) : \|T\| = n, B(T) \leq B, S(T) \subseteq S\}$ to be the minimum time required to broadcast a message to n processors, given the service providers S and budget B. Similarly, let $B(t,n,S) = \min\{B(T) : height(T) \leq t, \|T\| = n, S(T) \subseteq S\}$ be the minimal cost of a broadcasting to n processors within time bound t and using service providers S; and $N(t,B,S) = \max\{\|T\| : height(T) \leq t, B(T) \leq B, S(T) \subseteq S\}$ denote the maximal number of processors to which it is possible to broadcast a message within time t and budget B, using service providers S.

We consider the following related problems (assume $S(T) \subseteq S$ everywhere below):

[1] Relaxing the last requirement one can obtain MULTI_PORT (rather than SINGLE_PORT) model. Relaxing the bound on in-edges is meaningless, and allowing multiple setup edges makes sense only in some MULTI_PORT models with different and parallel setup times.

Broadcast on a budget: given n, S and budget B, compute the fastest schedule T to broadcast to n processors and satisfying the budget: $height(T) = H(n, B, S), B(T) \leq B, \|T\| = n$;

Broadcast with a deadline: given n, S and deadline t, compute the cheapest schedule T satisfying the deadline : $B(T) = B(t, n, S), height(T) \leq t, \|T\| = n$;

Broadcast with a deadline and a budget: given service providers S, budget B and deadline t, compute a schedule T to broadcast to the maximum number of processors $N(t, B, S)$ while satisfying both the deadline and the budget : $\|T\| = N(t, B, S), height(T) \leq t, B(T) \leq B$.

Given an algorithm to solve one of these problems, each of the other two can be solved by binary search.

Given a broadcast T, the receiving time of a processor p, denote by $t_T(p)$, is the minimum of t_v for all nodes that are associated with p. We say that a schedule T' is a *refinement* of T, denoted $T' \leq T$, if for every processor p of T its receiving time in T' is no worse than in T, namely $t_{T'}(p) \leq t_T(p)$.

We denote by T_v a sub-tree (sub-schedule) of T rooted in v.

W.l.o.g we assume that l and the q_is are integral (otherwise we multiply the q_is by a suitable integer).

We note here that for the sake of generality we consider S and l to be a part of the input and with unlimited size (in terms of n). However, the interesting practical applications will commonly have $l = O(1)$ as it is a fixed parameter of the corresponding system, and typically S will be a fixed set of service providers too, in which case the q_i's are all $O(1)$.

2.2 Properties of Schedules

In this section we notice some properties of optimal schedules that allows us to approximate the optimal schedule in some cases.

Intuitively, it seems that the faster messages should be used first to maximize the number of sending nodes as fast as possible. This intuition is only partly correct. For example, if $S = \{(c_1 = 2, q_1 = 1), (c_2 = 1, q_2 = 2)\}$ and $l = 1$, broadcasting to two processors with a budget $B = 3$ is possible in time 2, but only if the root sends a slow message first and the fast message second.

However, if we consider the messages along any single path from the root to a node, then the intuition is correct: We say that a path in a broadcast T is *monotone* if the providers on it are ordered in a non decreasing order; the faster providers come first. A schedule is called monotone if every path from a root to a node is monotone.

Claim 2.1 *Any schedule has a monotone refinement.*

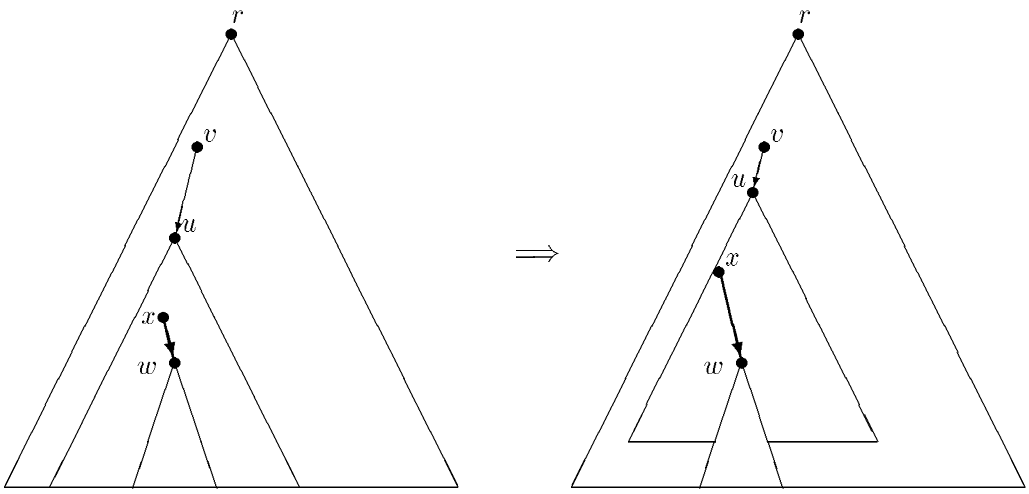

Figure 2: Fast transmission edges should precede the slow ones in any path. (see Claim 2.1). Nodes v, u, x, w all are on the same root-to-leave path, in both T and T'.

Proof:

Let schedule T be non-monotone. Then T has a root-to-leave path with a slower transmission edge $e_s = \vec{vu}$ preceding the faster $e_f = \vec{xw}$, $t_v < t_x$, $q_f < q_s$. Let T' be obtained from T by switching the providers of e_s and e_f (see Fig. 2). Then $t_v = t'_v$, $|t_w| = |t'_w|$. Similarly, the receiving times of the processors in T_w, T'_w and outside the subtree T_u, T'_u are the same. For any x in $T_u - T_w$, $t'_x = t_x - (q_s - q_f)$. Repeating the above, obtain the desired monotone schedule. ∎

In fact, the receiving times can be improved for some nodes (without delays for the others) by making sure that even on different root-to-leave paths, the faster edges are not preceded too much by the slower ones:

We say that a schedule T is *layered* if whenever transmission $e_f = \vec{vw}$ uses provider (c_f, q_f), and transmission $e_s = \vec{xy}$ uses (c_s, q_s) such that $q_s - q_f \geq l$, then $t_v < t_y$, namely, it cannot be that a fast edge will begin l units after a slow edge was ended.

Claim 2.2 *Every schedule has a layered refinement.*

Proof: Let T be a schedule with a faster edge $e_f = \vec{vw}$, a setup edge $\vec{vv'}$ and a slower edge $e_s = \vec{xy}$, such that $q_s - q_f \geq l$ and $t_y = t_x + q_s < t_v$. We prove the claim by showing that there is a schedule T' such that

- for each processor p in T_w its receiving time in T' is $t_{T'}(p) = t_T(p) - (t_v - t_x - q_s)$;

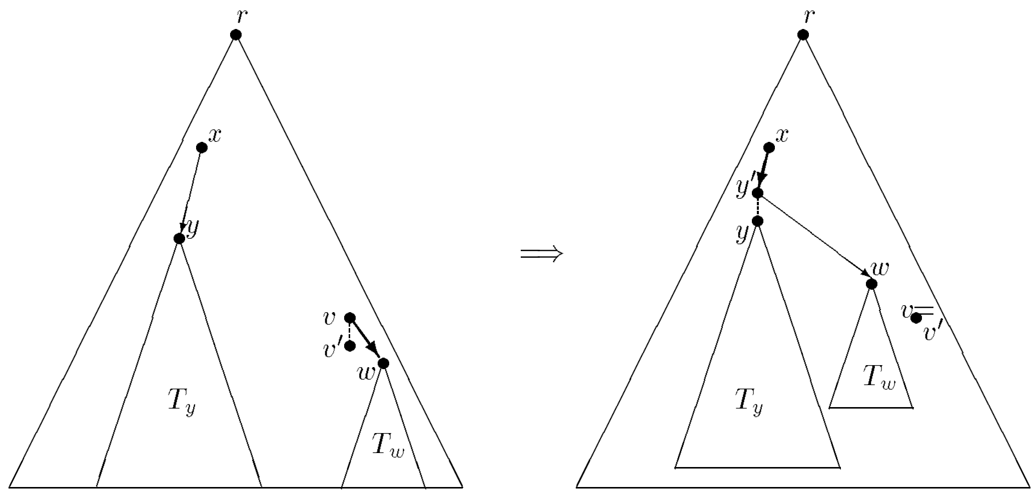

Figure 3: Switching fast and slow edges for Claim 2.2.

- for each p in T_y but not in T_v its receiving time in T' is $t_{T'}(p) = t_T(p) - (q_s - q_f - l)$;
- for each p in $T_{v'}$ but not in T_y its receiving time is $t_{T'}(p) = t_T(p) - l$;
- for each p in both $T_{v'}$ and T_y its receiving time is $t_{T'}(p) = t_T(p) - (q_s - q_f - l) - l$;
- for all other processors the receiving times in T, T' are the same.

Thus, if $q_s \geq q_f + l$ and $t_v - t_x \geq q_s$ then T' is a refinement of T. By repeating the above till there are no such e_s, e_f obtain the layered refinement of T.

Consider the effect of changing T to T' as shown in Fig. 3: Slow transmission edge $\vec{xy}$ is replaced with a fast one $\vec{xy'}$ followed by a setup $\vec{y'y}$, w now receives a slow message from y' (that is why a setup edge $\vec{y'y}$ is added); since v no longer sends a message (to w) its setup edge $\vec{vv'}$ is no longer needed, so v' can be moved up into v; the rest is unchanged.

Clearly, the nodes/processors outside T_y and T_v are not affected by the change from T to T'.

$t'_w = t_x + q_f + q_s$; $t_x = t_w - q_f - (t_v - t_x)$. So, $t'_w = t_w - t_v + t_x + q_s$. Thus, for every z in T_w, $t'_z = t_z - (t_v - t_x - q_s)$.

$t'_y = t_x + q_f + l = t_y - q_s + q_f + l$. Thus, for any z in T_y (but not in T_w, for which the above takes precedence), $t'_z = t_z - q_s + q_f + l$.

Finally, the result of moving v' up l steps is that all its children have their time improved by l (if v' is in T_y this saving is added to the above). ∎

So, w.l.o.g. we can assume that all schedules are monotone and layered.

3 Algorithms

We first use the properties that where discussed in the previous section in order to develop a fast approximation algorithm for the broadcast problem of two providers. Then we consider the general case.

3.1 Broadcasting with two service providers

Let $S = \{(c_f, q_f), (c_s, q_s)\}$ and assume that $q_s \geq q_f + l$. When the number of service providers is only two it is easy to compute how many times each provider should be used for an optimal broadcast on a budget schedule: Let f be the number of expensive transmissions and let s be the number of cheap transmission. Then

$$f \cdot C_f + s \cdot C_s \leq B \qquad \text{and} \qquad f + s = n.$$

Substituting $s = n - f$ into the inequality we get

$$f \cdot C_f + (n - f) \cdot C_s = f \cdot (C_f - C_s) + n \cdot C_s \leq B$$

so we want the largest integral f such that

$$f \leq \frac{B - n \cdot C_s}{C_f - C_s}$$

and for simplicity, we assume in the sequel that equality holds in the above equation.

We present now a linear time algorithm to obtain a *nearly optimal* schedule. Denote by $H(m, q) = H(m, mc, \{(c, q)\})$ the optimal broadcast time to m nodes using a single service provider with the delay q. An optimal single provider schedule can be computed in linear time as in [8, 9].

Consider an optimal single provider schedule T^f for broadcasting a message to f processors using the fast provider only. Its height is $height(T^f) = H(f, q_f)$ and it can be constructed (in linear time) by the algorithm of [8, 9]. Using the same algorithm, we construct an optimal schedule T^s of height $height(T^s) = H(\lceil \frac{s}{f+1} \rceil, q_s)$, for broadcasting to $\lceil \frac{s}{f+1} \rceil$ processors using only the slow provider. The final schedule T is obtained by appending a copy of T^s to each of $f+1$ leaf of T^f (so that each leaf of T^f becomes a root of a copy of T^s; some resulting nodes will be unused). Then $height(T) \leq H(f, q_f) + H(\lceil \frac{s}{f+1} \rceil, q_s)$. We show that an optimal schedule can not be much better:

Claim 3.1 *Let n, S, B, f, s be defined as above. Then $H(n, S, B) > H(f, q_f) + H(\lceil \frac{s}{f+1} \rceil, q_s) - (q_s + q_f)$.*

Proof: By Claim 2.2 there exists a layered optimal schedule T such that if $\vec{vw}$ is the latest fast and $\vec{xy}$ the earliest slow message, then $t_v - t_x < q_s$. Also, clearly $t_w \geq H(f, q_f)$, so $t_v \geq H(f, q_f) - q_f$. Combining these we get $t_x > t_v - q_s \geq H(f, q_f) - (q_f + q_s)$.

W.l.o.g., there are $\lceil \frac{s}{f+1} \rceil$ processors in T_x. So, $height(T_x) \geq H(\lceil \frac{s}{f+1} \rceil, q_s)$, and therefore $height(T) = H(n, S, B) > H(f, q_f) - (q_f + q_s) + H(\lceil \frac{s}{f+1} \rceil, q_s)$. ∎

Thus, schedules within $q_s + q_s$ from optimal can be computed with only two applications of algorithms of [8, 9].

3.2 General Broadcast

In this section we present algorithms to compute *optimal* schedules for the broadcast on a budget and broadcast with a deadline. Let $|S| = k$, i.e. there are k service providers. Since S remains fixed throughout this section, for the sake of simplicity we omit it from all the notations.

Clearly, the cost of the cheapest possible broadcast is $c_k n$ (by using only the kth cheapest provider), with time equals $H(n, q_k)$. While the fastest broadcast can be done in $H(n, q_1)$ using only the fastest service provider.

We now present our first algorithm, $A1$, that computes $B(t, n)$ and the corresponding schedule in $O(kn^2(nl + q_k))$ steps. This time is polynomial in n, k (this is expected since the output size is $\Omega(n)$). However, it is proportional to q_k (and $l \leq q_k$) which is exponentially larger than its description. We then show how to improve algorithm $A1$ to become polynomial in the description of q_k too.

The algorithms use dynamic programming based on the following claim.

Claim 3.2 *For every $m \geq 0$ and every t,*

$$B(t, m) = \infty \quad \text{for} \quad t < H(m, q_1)$$

$$B(t, 0) = 0 \quad \text{for} \quad t \geq 0 = H(0, q_1)$$

$$B(t, m) = min_{1 \leq i \leq k} \ min_{1 \leq r \leq m-1} \ [c_i + B(t - q_i, r - 1) + B(t - l, m - r)] \tag{1}$$

Proof: The first two equalities are by convention. For the third, any broadcast to m processors must start with sending a message using one of the k service providers, after which the receiving processor will broadcast to some other $r - 1$ processors. In the meantime, the origin is free to take care of the remaining $m - r$ processors, starting after l steps (which is its setup time). ∎

Algorithm $A1$.

for $m = 1, \ldots, n$

  for $t = H(m, q_1), \ldots, H(m, q_k)$ compute B(t,m) using Equation (1) .

The corresponding schedule T can be obtained by straight-forward adjustments of the above algorithm.

Claim 3.3 *Algorithm* $A1$ *computes* $B(t,n)$ *in* $O(kn^2(nl+q_k)\lg q_k)$ *steps.*

Proof: Each application of Equation (1) takes $O(kn \lg q_k)$ steps ($\lg q_k$ is for the numerical computations involving q_i). In order to compute $B(t,m)$ over all the range, no more than $n \cdot (q_k + nl)$ such application are needed. ■

To improve the algorithm complexity to be polynomial in $\lg q_k$ (rather then in q_k) we note that $B(t,m)$ need not be computed for any t in the range $q_{k-1} + (m-1)l < t < q_k$. The reason for this is that when $t < q_k$ the kth provider can not be used at all. Thus, the cheapest possible cost will be $c_{k-1}m$, which is obtained for $t = q_{k-1} + (m-1)l$. Namely, $B(t,m) = c_{k-1}m$ for each t in the above range.

A similar argument applies to t in the range $q_i + (m-1)l < t < q_{i+1}$, for each $i = 1, ..., k-1$:

Claim 3.4 *For each* $i = 1, ..., k-1$, *for each* t, $q_i + (m-1)l < t < q_{i+1}$
$B(t,m) = c_i m$

We get the following improved algorithm: **Algorithm** $A2$

$for\ i = 1, ..., k$
  $for\ t = q_i i, ..., q_i i + (n-1)l$
    $for\ m = 1, ..., n$ *compute* $B(t,m)$ *using Equations* (1) *and* (3.4).

Claim 3.5 *For input* $S = \{(c_i, q_i) | i = 1, ..., k\}, l,\ n \geq 1$ *and* t, $1 \leq t \leq q_k + (n-1)l$, *Algorithm* $A2$ *computes* $B(t,n)$ *in* $O(k^2 n^3 l \lg q_k)$ *steps.* ■

We note here that the dependency in l is not polynomial in the length of its description - this is indeed unsettled here, although for practical purposes l should be taken as fairly small, usually $O(1)$ (see also Sec. 3.3).

3.3 Complexity issues

We have shown that for input $S = \{(c_i, q_i) | i = 1, ..., k\}$ representing the service providers, l representing the setup time, and integers B, n, representing a total budget and number of targets respectively, algorithm $A2$ takes $O(k^2 n^3 l \lg q_k)$ steps. This time bound is polynomial in the input length only if n is given in unary and $l = O(max(n, \lg q_k))$. We can't expect a polynomial algorithm whose dependency in n is $poly(\lg n)$ as the output size (the broadcast tree) is of size $\theta(n)$. However, the following is an associated decision problem whose complexity is unsettled.

The broadcast decision problem: Given a set S of k service providers, budget B, a set-up delay l, the number of processors n and the deadline H, is there a broadcast schedule T on n nodes with $height(T) \leq H, B(T) \leq B$ (i.e., can we broadcast to n processors within time H and budget B?) Corresponding optimization problems would be to compute the functions $B(), H(), N()$.

We assume that $l \leq q_1$ as we take the setup time to be included in the delay and may always assume that $q_i \leq H$ and $c_i < B$, for all i (otherwise the ith provider cannot be used and so can be ignored). Hence the length of the input can be taken to be $poly(k, \lg H, \lg B, \lg n)$ and algorithm $A2$ does not provide a poly-time algorithm. Few remarks are due here.

- For $k = 1$ (no optimization is needed) there is a poly-time algorithm for the broadcast decision problem [8, 9].
- For $k > 1$ and general set of providers, S, we do not know any upper bounds for the complexity of the broadcast decision problem except exponential-time even for $k = 2$.

For the restricted problem in which S is such that $q_k = O(poly(\lg n))$ the broadcast decision problem is in NP as stated by the following theorem:

Theorem 3.6 *Let $S = \{(c_i, q_i) \| i = 1, \ldots, k\}$ be a set of service providers, l the setup time, and B, n, H, representing a total budget, number of targets, and deadline respectively. Assume that $l \leq q_k = O(poly(\lg n))$. Then the broadcast decision problem is in NP.*

The intuition behind the proof is the following: We first show that if the service providers time is relatively small ($q_k = O(poly(\lg n))$) then there is a shallow broadcast tree (this is shown in Lemma 3.7). Next we show that any shallow broadcast tree can be witnessed (and verified) by examining a relatively small amount of information, namely proportional to the tree depth rather than to the tree size (this is shown in Lemma 3.8). The reason for this is that to witness a schedule, the tree structure is of no importance - the only important data is how many service providers of each type are being used at each point in time. However, this data is proportional to the tree depth rather than to its size.

We now present the formal proof of the theorem.

Proof: The proof will follow from the Lemmata below.

Lemma 3.7 *Let $S = \{(c_i, q_i) \| i = 1, \ldots, k\}, q_k = O(poly(\lg n))$. Then, $H(n, B, S) = O(poly(\lg n))$.*

Lemma 3.8 *Let $S = \{(c_i, q_i) \| i = 1, \ldots, k\}$, and l, B, n, H as before. Then a broadcast tree T of size n and $B(T) \leq B, height(T) \leq H$, exists iff there are k sequences of non-negative integers*

$\{e_t^i\}_{t=0}^{t=H}$, $i = 1, ..., k$ *(representing the number of edges (service providers) of type i at time t), two sequences $\{n_t\}_{t=0}^{t=H}$ and $\{o_t\}_{t=0}^{t=H}$ (that represent the number of new processors and available processors at time t) that satisfy the following conditions:*

1. $n_0 = 0, \quad o_0 = 1$ *- initial conditions.*
2. $n_t = \Sigma_{i=1}^{i=k} e_{t-q_i}^i$ *- new nodes at time t.*
3. $o_t = o_{t-l} + n_{t-l}$ *- nodes become available again after l steps of setup time.*
4. $\Sigma_{i=1}^{i=k} e_t^i \leq n_t + o_t$ *- number of transmissions at time t.*
5. $1 + \Sigma n_i = n$ *- total number of processors.*
6. $\Sigma_{t=0}^{t=H-1} \Sigma_{i=1}^{i=k} c_i e_t^i \leq B$ *- total budget.*

The theorem follows, as from the assumptions of Lemma 3.7 there is a valid broadcast schedule T of time $H' = O(poly(\lg n))$. A Non-deterministic procedure to verify the existence of such schedule T is to guess H', k sequences $\{e_t^i\}_{t=0}^{t=H'}$, $i = 1, ..., k$ (representing the number of edges (service providers) of type i at time t), two sequences $\{n_t\}_{t=0}^{t=H'}$ and $\{o_t\}_{t=0}^{t=H'}$ (that represent the number of new processors and available processors at time t) and to verify that the conditions of Lemma 3.8 are met (which can be done in $O(poly(\lg n, k, \lg B))$ time). ■

We now proceed to prove the Lemmata.

Proof (of Lemma 3.7):

Given $S = \{(c_i, q_i) \| i = 1, \ldots, k\}$, n, and a budget B. Let T be an optimal broadcast schedule; we prove that $height(T) = poly(\lg n)$ which implies the lemma. Let T' be an optimal broadcast schedule for for n processors that uses only q_k. Certainly $height(T) \leq height(T')$. However, $height(T') = O(poly(\lg n))$ [3, 9]. ■

Proof (of Lemma 3.8):

The conditions of the Lemma attempt to characterize a schedule T for inputs S, B, n that ends within time H. We first explain the meaning of the conditions in the Lemma. In general, these conditions assert that each level (time point) in the schedule is consistent, based on the consistency of previous levels. This fact, together with a consistent starting/ending level, and a total time/budget as required, is enough to assert the existence of the schedule.

Indeed, n_t represents the number of new nodes at time t, namely these nodes that correspond to processors that have received the message at time t. These nodes are available to immediately start transmitting the message to other nodes. o_t represents the number of available old nodes at time t; namely, these nodes that have received the message at time $t - i \times l$ for some i (we assume here that such nodes transmit whenever possible, that is: every l units of time). There are other nodes that might be in the middle of the setup process for their next message, but these are not involved in any action at time t, and will not be accounted for. e_t^i represents the number of service providers of type i that are used at time t. Given this, Conditions 1-4 assert the consistency of the schedule tree: Condition 1 is the initial situation: there is a single node which is available at time $t = 0$. Condition 2 relates the edges to the nodes; it asserts that with every provider e_t^i there is a node (at the end of this edge) at time $t + q_i$. Condition 3 states that the available old nodes at time t are exactly the old and new nodes of time $t - l$ (for which the setup time is just over). Condition 4 relates the available nodes at time t to service providers used at time t, namely, available nodes to send messages at time t are exactly the old and new nodes at that time.

Conditions 5 and 6 assert the resource bounds of the schedule: Condition 5 asserts that the schedule is a broadcast tree for n nodes; Condition 6 asserts that the total cost of the schedule is bounded by B.

We now formally prove the Lemma:

The only-if part is obvious: Assume that there is a broadcast schedule T on n nodes that meets the budget and time. We call a node v in the schedule *new* if it is the end of an edge that is marked by some provider, otherwise we call it *old*. We may assume that each node sends the message as soon as he can (otherwise, the tree can be obviously transformed into another optimal tree which meets this assumption). Let n_t, o_t be the number of new and old nodes at time t. Let $e_t^i, \;\; i = 1, ..., k$ be the number of calls of the ith service provider that are used at time t. Then it is easy to see that the conditions of the lemma hold.

For the other direction, assume that the k sequences $\{e_t^i\}_{t=0}^{t=H}, \;\; i = 1, ..., k$ (representing the number of edges of type i at time t), and the two sequences $\{n_t\}_{t=0}^{t=H}$ and $\{o_t\}_{t=0}^{t=H}$ (representing the new and old nodes) are given so that Conditions 1-6 are met for time $t = 0, 1, ..., H$. We will show how to construct a schedule T for the given input, of time at most H and budget B. The construction is by induction on H, the length of the sequences. For $H = 0$ the construction is trivial (with $n = 1$). Assume that for every set of sequences of length $H - 1$ or less that meets the condition of the Lemma, there is a broadcast schedule of time bounded by $H - 1$ (and appropriate n as is implied by Condition 5). We will prove it for H. Define $n'_t = n_t, \;\; o'_t = o_t$ for $t = 0, ..., H - 1$ and $f_t^i = e_t^i$ if $t + q_i \leq H - 1$ and 0 otherwise. These new variables certainly satisfy the conditions of the lemma for $t = 0, ..., H - 1$ for $n' = n - n_H$ and $B' = B - \Sigma_{i=1}^{i=k} e_{H-q_i}^i$. Thus, by induction, there is a broadcast schedule T' for n' of time bounded by $H - 1$ and budget B'. Now we want to add back the $\Sigma_{i=1}^{i=k} e^i_{H-q_i}$ edges that we set to 0 to construct a broadcast schedule T on n nodes. To do that we need to allocate $e^i_{H-q_i}$ nodes at time $H - q_i$ in T' and add one edge to each, marked with the ith provider. We can do this only if there are indeed $e^i_{H-q_i}$ nodes that can be allocated. Namely, these nodes in T' must have no outgoing edges. However, Condition 4 guarantees that for $t = H - q_i$, $n'_t + o'_t = n_t + o_t \geq \Sigma_{j=1}^{j=i} e^j_t = \Sigma_{j=1}^{j=i-1} f^j_t + e^i_t$ which guarantees the right amount of nodes needed. Of course, each such node comes with a new node at its endpoint which is at time bounded by H. This results in a broadcast schedule T of time H, budget B, and n nodes. ∎

4 Conclusion

Many "shopping problems" can be formalized in the context of the model presented in this paper. In the context of communication service providers, we gave an algorithm which computes an optimal broadcast schedule.

There are still many related open questions. In particular the complexity of the broadcast decision problem is not completely settled. A different direction is to enrich the model by introducing probabilities of success, dealing with randomized strategies, varying service loads, and other real-world parameters. This direction will result in a "multi-(parameter,cost)" optimization problems.

References

[1] A. Bar-Noy, J. Bruck, C. Ho, S. Kipnis, and B. Schieber. Computing global combine operations in the multi-port postal model. *IEEE Transactions on Parallel and Distributed Systems*, 6(8):896–900, August 1995.

[2] A. Bar-Noy and S. Kipnis. Multiple message broadcasting in the postal model. In *Proc. 7th International Parallel Processing Symposium*, pages 463–470. IEEE, April 1993.

[3] A. Bar-Noy and S. Kipnis. Designing broadcasting algorithms in the postal model for message-passing systems. *Mathematical Systems Theory*, 27(5):431–452, 1994.

[4] A. Bar-Noy, S. Kipnis, and B. Schieber. Optimal computation of census functions in the postal model. *Discrete Applied Mathematics*, 58:213–222, 1995.

[5] J. Bruck, R. Cypher, and C.T. Ho. Multiple message broadcasting with generalized Fibonacci trees. In *Proc. 4th Symposium on Parallel and Distributed Processing*, pages 424–431. IEEE, December 1992.

[6] J. Bruck and C.T. Ho. Efficient global combine operations in multi-port message-passing systems. *Parallel Processing Letters*, 3(4):335–346, December 1993.

[7] J. Bruck, C.T. Ho, S. Kipnis, and D. Weathersby. Efficient algorithms for all-to-all communications in multi-port message-passing systems. In *Proc. 6th Annual Symposium on Parallel Algorithms and Architectures*, pages 298–309. ACM, June 1994. To appear in *IEEE Transactions on Parallel and Distributed Systems.*

[8] S.-N. Choi and M. Golin. Lopsided trees: Algorithms, Analyses and Applications. In *Proc. of the 23rd Intl. Colloquium on Automata Languages and Programming*, 1996.

[9] M. Golin and A. Schuster. Optimal Point-to-Point Broadcast Algorithms via Lopsided Trees. In *Proc. 5th Israeli Symp. on Theory of Computing and Systems*, pages 63–73, 1997. To appear in Disc. Appl. Math.

[10] O. Etzioni and S. Hanks and T. Jiang and R.M. Karp and O. Madani and O. Waarts. Efficient Information Gathering on the Internet. In *Proc. 37th Symp. on Foundations of Computer Science*, pages 234–243, 1996.