Finding Most Sustainable Paths in Networks with Time-Dependent Edge Reliabilities

Goran Konjevod[1], Soohyun Oh[1], and Andréa W. Richa[1]

Department of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287-5406, USA
{goran, soohyun, aricha}@asu.edu

Abstract. In this paper, we formalize the problem of finding a routing path for the streaming of continuous media (e.g., video or audio files) that maximizes the probability that the streaming is successful, over a network with nonuniform edge delays and capacities, and arbitrary time-dependent edge reliabilities. We call such a problem the *most sustainable path (MSP)* problem. We address the MSP problem in two network routing models: the wormhole and the circuit-switching routing models. We present fully-distributed polynomial-time algorithms for the streaming of constant-size data in the wormhole model, and for arbitrary-size data in the circuit-switching model. Our algorithms are simple and assume only local knowledge of the network topology at each node. The algorithms evolved from a variation of the classical Bellman-Ford shortest-path algorithm. One of the main contributions of this paper was to show how to extend the ideas in the Bellman-Ford algorithm to account for arbitrary time-dependent edge reliabilities.

1 Introduction

Continuous media applications — such as audio and video applications — became standards in today's wide-area internetworks. It is expected that by 2003, continuous media will account for more than 50% of data available on the web servers [6], [8]. When streaming such an application through the network, it is desirable not to make flow stops to correct errors nor to resend lost packets. Since current networks are not fully reliable, we would like to ensure QoS guarantees by finding a path in the network with the lowest probability of failure for the entire streaming procedure. Conventional routing algorithms for connection-oriented traffic tend to use shortest paths, without taking into consideration how reliable the paths are. A problem in finding a path to be used in a long duration streaming application lies on how to properly capture and handle the relevant dynamic changes of the network conditions.

The reliability of a link (or edge) in the network is a function of many parameters — such as network traffic pattern, link bandwidth, routing paths' latency and congestion — which vary with time. Thus, when estimating how reliable a path is for a streaming application, one needs to account for the reliabilities of all edges involved in the streaming procedure, over all time steps. Moreover, estimating the reliability of a path for a streaming application must account not only for the time-dependent reliabilities of the edges used, but also for the congestion of the path and the size of the data: The size of the message over the capacity of the path chosen indicates the number of time steps each edge in the path will be used during the transmission.

More specifically, we consider the problem of finding a path for the continuous transmission (streaming) of a message of size σ (i.e., consisting of σ units of data) from a given source node s to a given destination node d, in a network with arbitrary polynomial (on the number fo network nodes) edge capacities and delays, and with time-dependent edge reliabilities. Since a message consists of multiple units of data, the duration of the connection between s and d for transmitting the message may be much longer than a short time interval. Thus we define the *sustainability* of a path with respect to transmitting a message to be the probability that all the edges used in the transmission at a given time step will be alive during all time steps used for the transmission. Note that the only parameter of the data relevant to computing the sustainability of a path is the size of the message. Hence we will, in the remainder of this paper, talk about most sustainable paths in a network for a given message size.

We consider two different network routing models [13]: wormhole routing and circuit-switching.

In **wormhole routing**, a message is a sequence of fixed-size packets called *flits*. The first flit is called the head and the rest the body of the message. The flits are sent in a pipelined fashion. Only the head of a message has the routing information, and the body follows the head contiguously. Since the message moves flit by flit, no buffering is required. This model works on top of a packet-forwarding network model, thus only the edges that are actively transmitting packets at any given time step need to be alive during that time step.

In the **circuit-switching model**, we reserve a connection path between the source and destination before the transmission begins. Once the transmission is initiated, the message is never blocked. Since the edges in the path are reserved, no buffering is necessary. In this model, which is connection-oriented, all the edges of the reserved path must be alive throughout the entire transmission of the message, otherwise the transmission will fail. The popular Asynchronous Transfer Mode (ATM) networks [7], [10] provide circuit-switching network services on a packet-switching environment via virtual circuits.

One problem that can occur in the wormhole routing model is called the *head blocking* deadlock: If the head of a message cannot advance because it competes for the traversal of an edge with some other flits in the body of the message, then all the flits must stall, and the transmission of the message is halted. This scenario occurs when there exists a cycle in the selected routing path which is not long enough to accommodate the entire message: i.e., the length of the cycle is less than the size of the message over the capacity of the path. This problem is avoided in the circuit-switching model, because a most sustainable path in this model cannot contain cycles.

In any continuous media streaming application, it is important that the data to be transmitted arrives at the destination node in the order it left the source node (we call this property the *nonpassing property*). Both the wormhole and the circuit-switching model guarantee that this property is satisfied.

In Section 3, we propose a polynomial time algorithm to compute a most sustainable path for the transmission of constant-size messages in the wormhole routing model. We call such a path a σ*-most sustainable path* from s to d, where σ is the size of the message to be transmitted, or σ*-MSP* for short. In our wormhole routing scenario, no waiting or buffering at a node is allowed once the transmission starts. In Section 4, we propose a polynomial time algorithm to compute a most sustainable path for the transmission of a message of an arbitrary size σ in the circuit-switching model. We call such a path a σ*-most sustainable reserved path* from s to d, or a σ*-MSRP* for short.

To the best of our knowledge, this is the first work that handles time-dependent edge reliabilities when computing the best path for data streaming. All the algorithms we present are simple and fully distributed. They can be implemented using only local knowledge of the network at the nodes. (Each node only needs information about its local topology: the edges adjacent to it and their capacities, delays, and reliabilities.)

We assume that each node has full knowledge of its edge reliability function. We understand that this may not be strictly true in a real setting. In a more realistic scenario, these functions would be estimated using prior traffic data. Also, in a realistic setting, we can assume the reliability of each edge to be at least $\frac{1}{f(n)}$ and no more than $1 - \frac{1}{f(n)}$, for some (positive) polynomially increasing function f on the number of nodes in the network n. Without this assumption, we could have most sustainable paths of exponential total delay, even for short messages, which would result in an unreasonable over-utilization of network resources.

The algorithms evolved from the distributed Bellman-Ford shortest path algorithm, (which can also be used directly to compute most sustainable paths for unit-size messages). Our main contribution is to show how the Bellman-Ford framework can be extended in order to account for time-dependent edge reliabilities. Also, we introduce pruning techniques for incremental construction of most sustainable subpaths.

The remainder of the paper is organized as follows. In the next section, we give a brief literature review of related network routing problems. The description of the MSP problem is presented in Section 3, as well as a polynomial-time algorithm for solving this problem for constant size data. In Section 4, we define the MSRP problem and propose an algorithm for solving this problem in polynomial time. In Section 5 we conclude and discuss future work.

2 Related work

Over the last few decades, routing algorithms have been the subject of extensive research. The shortest-path routing problem has certainly been the most widely studied routing problem. Various shortest-path algorithms for networks with nonnegative edge delays have been proposed and implemented, the best-known being Dijkstra's algorithm and the Bellman-Ford algorithm [3]. Both these algorithms satisfy the optimal subpath property, which says that any subpath of an optimal path is also optimal.

One variation of the shortest path problem is finding a path to send a message of size σ from a source to a destination with minimal transmission time in a network with nonuniform edge delays and capacities [4]. The most important observation here is that due to the variance in edge capacities, the transmission time is closely related to the size of the message. Another variation, closer to the problems we study, is finding a most reliable path for a single packet [1].

Tragoudas [15] was the first to formalize the problem of finding a most sustainable path for an arbitrary size message in a network with nonuniform edge delays, capacities, and reliabilities. He presented an *off-line, centralized* algorithm for this problem. However, his result assumes that all the network parameters, including edge reliabilities, are *static*, that is, do not vary over time.

The shortest-path problem in networks in which the delays of the edges change with time was studied by Orda and Rom [11], [12]. In [11], they present algorithms and hardness results for the problem of finding shortest-paths under various *waiting constraints.* In their network model, the *nonpassing property* is violated. (The nonpassing, or FIFO, property says that two packets that traverse an edge (u, v) will arrive at node v in the same order as they left node u.) In network streaming applications, the nonpassing property should be preserved. Therefore, the results of Orda and Rom, in spite of being closely related to our problem, cannot be extended to solve the MSP problem.

A new model, named the *flow speed model*, for time-dependent networks where the flow speed of each link depends on the time interval is suggested by Sung, Bell, Seong, and Park [14]. They use a variation of Dijkstra's algorithm, incorporating a procedure for calculating the link travel time from the flow speed so that their model satisfies the nonpassing property.

Cooke and Halsey [5] modified the Bellman-Ford algorithm to solve a shortest-path problem in networks with time-dependent internodal transit times.

Recently, Miller-Hooks and Patterson [9] proposed a pseudopolynomial algorithm for solving the integral time-dependent quickest flow problem. In their network model, a packet may achieve an earlier arrival time at the destination by waiting at a node while en route, which may violate the nonpassing property.

In [2], various routing problems in time-dependent labeled networks with applications to transportation science are described. A realistic model is given in which Dijkstra's algorithm can be extended to time-dependent edge-delays. (The nonpassing property is essential here.)

Another recently studied routing problem is finding an acceptable path under multiple constraints to meet quality of service (QoS) requirements for the correct processing and transmission of continuous media on integrated communication networks (for example, ATM networks) [16], [17]. In the current routing approaches for connection-oriented network services, routing decisions are made using only information on the network conditions at connection establishment.

Since continuous multimedia streams are comparatively long-lived, each routing decision should consider the network conditions over a longer time period.

3 The MSP Problem

In this section, we study the problem of finding most sustainable paths in the wormhole routing model. We first introduce the model and related definitions.

Consider a network $G(V,E)$, where V is the set of nodes and $E \subseteq V \times V$ the set of edges. Let $n = |V|$. Associated with each edge (i,j), we have a delay ℓ_{ij} and a capacity c_{ij}, as well as a time-dependent reliability function $r_{ij}(t)$. (whose value at time t is the probability that the edge (i,j) operates correctly at that time step.) As mentioned in the introduction, we assume that $\frac{1}{f(n)} \leq r_{ij}(t) \leq 1-\frac{1}{f(n)}$, for some polynomially increasing f. The edge delay ℓ_{ij} denotes the number of time steps required for a unit of data (or *packet*) to be transmitted from i to j through edge (i,j), and the capacity c_{ij} gives the number of units of data that can traverse (i,j) in a single time step. We assume that edge delays and capacities are positive polynomials in n. Without loss of generality, we assume that all capacities and delays are integral and at least 1.

In the wormhole routing model we need only to account for the reliability of an edge at time t if a packet is traversing the edge at that time. The sustainability of a path is thus defined as follows. Let $c(P)$ denote the *capacity of path* P, the smallest capacity of an edge in P. Let P^{ℓ}_{vw} be a path from node v to node w of length ℓ, where the length of a path is equal to the sum of the delays of the edges in the path. Let $P^{\ell}_{v_0 v_k} = (v_0, v_1, \ldots, v_k)$. Let ℓ_j, $j = 1, \ldots, k$, denote the length of the subpath of $P^{\ell}_{v_0 v_k}$ from v_0 to v_j, and let $\ell_0 = 0$. For convenience, we assume that v_0 begins transmission at time zero. If $c = \lceil \sigma / c(P^{\ell}_{v_0 v_k}) \rceil$, the sustainability of the path $P^{\ell}_{v_0 v_k}$ for a message of size σ is given by

$$Re^{\sigma}(P^{\ell}_{v_0 v_k}) = \prod_{i=1}^{k} \prod_{t=\ell_{i-1}}^{c+\ell_i-1} r_{v_{i-1} v_i}(t).$$

The sustainability of an edge (v_i, v_{i+1}) for this transmission is

$$r^{\sigma}_{v_{i-1} v_i}(\ell_{i-1}) = \prod_{t=\ell_{i-1}}^{c+\ell_i-1} r_{v_{i-1} v_i}(t). \tag{1}$$

Now the path sustainability can be rewritten as

$$Re^{\sigma}(P^{\ell}_{v_0 v_k}) = \prod_{i=1}^{k} r^{\sigma}_{v_{i-1} v_i}(\ell_{i-1}). \tag{2}$$

We use $Re(P^{\ell}_{v_0 v_k})$ to denote the case when $\sigma = 1$.

For a fixed length ℓ, a path P^{ℓ}_{vw} from v to w which maximizes $Re^{\sigma}(P^{\ell}_{vw})$ is called a *most sustainable path of length* ℓ from v to w for a message of size σ, or for short, a (σ, ℓ)–MSP from v to w. A σ–MSP from v to w is the path that has maximum sustainability among all the (σ, ℓ)-MSPs from v to w for all possible lengths ℓ.

In the wormhole routing, there are no buffers at the nodes. Thus packets must travel contiguously. In case of only one packet (i.e., $\sigma = 1$), our problem is similar to that of finding a shortest path with time-dependent edge delays, where no waiting is allowed [11]. Orda and Rom show that the optimal subpath property is not satisfied in their model and claim NP-hardness of this problem. However, in our problem there exists an optimal subpath property (to be defined in Section 3.1), which allows dynamic programming to find an optimal solution.

3.1 The MRP algorithm: unit-size messages

We first consider unit-size messages. The ideas presented for this simple case will be generalized in the next section. The algorithm is similar to Bellman-Ford [3]. A most sustainable path for a single unit of message is also called a *most reliable path* (MRP). For a fixed ℓ, a path P_{sv}^{ℓ} from s to v which maximizes $Re(P_{sv}^{\ell})$ is called a *most reliable path of length* ℓ, or for short ℓ-MRP, from s to v. In the remainder of this paper, we will use $\overline{P}_{sv}^{\ell}$ to denote an ℓ-MSP path from s to v.

The source node s initiates the algorithm by informing each of its neighbors v of the reliability of the ℓ_{sv}-MRP which contains edge (s, v) only.

Upon the receipt of the sustainabilities of paths of length ℓ from s to v from its neighbors, node v chooses the one $(\overline{P}_{sv}^{\ell})$ with maximum sustainability. Then v computes the sustainabilities of paths composed of $\overline{P}_{sv}^{\ell}$ followed by each of the edges adjacent to v. The algorithm terminates once the destination node d has computed all the values $Re(\overline{P}_{sd}^{\ell})$ for all possible MRP lengths ℓ. The path with the largest sustainability is then an MRP from s to d. The algorithm keeps track of all possible MRP lengths from s to each node v in G. If there exist two or more paths of the same length, the algorithm keeps the most sustainable one. Any ties are broken arbitrarily.

Each node v maintains a set of tuples of the form (ℓ, R, u) where for some path P from s to v, ℓ is the length and R the sustainability of P, and u is the node preceeding v in P. At each time step ℓ, each node v updates a set $L(v)$ whose elements are tuples corresponding to the k-MRP from s to v, for all possible values of k where $k \leq \ell$ (if such a path exist). Node v chooses the ℓ-MRP based upon learning of a tuple $(\ell, R, \cdot)$ from its neighbors and it ignores the others. Let the chosen tuple be (ℓ, R, u). Then v adds the tuple (ℓ, R, u) to its set $L(v)$. Node v calculates $Re(P_{sw}^{\ell+\ell_{vw}})$, the sustainability of the path of length $\ell + \ell_{vw}$ composed of the path from s to v associated with the tuple (ℓ, R, u) followed by edge (v, w). The sustainability of path $P_{sw}^{\ell+\ell_{vw}}$ is given by

$$Re(P_{sw}^{\ell+\ell_{vw}}) = R \times \prod_{t=\ell}^{\ell+\ell_{vw}-1} r_{vw}(t).$$

An upper bound M of the number of possible MRP lengths in the network can be computed as follows. It is easy to see that there cannot be any MRP of length greater than $M = n + x$ where x is maximum y such that $\frac{1}{f(n)} \leq (1 - \frac{1}{f(n)})^y$, (where $f(n)$ is the polynomial function that bounds the edge reliabilities). Since $\frac{1}{f(n)} > 0$, $\frac{1}{f(n)} \leq e^{-\frac{y}{f(n)}}$. By solving for y, we get

$$y \leq f(n) \ln f(n).$$

Hence M is bounded by a polynomial on n.

To analyze the running time, note that each tuple calculation and comparison takes constant time. Since in the worst case all nodes are adjacent (and all edge lengths are the same), at each time step, every node receives at most n tuples. Hence the new tuple computations and comparisons take $O(n)$ time. Since $L(d)$ will have at most M tuples (one per time step), choosing the tuple with maximum sustainability at time step $M+1$ takes $O(M)$ time and the overall time bound is $O(nM)$.

In our MRP algorithm, each node v compares the tuples of the form $(t, \cdot, \cdot)$, at time step t only, in order to choose a t-MRP from s to v. Hence if a tuple with associated length t arrives at node v at time t' other than t, the algorithm cannot guarantee that it finds a t-MRP. Therefore it is important that all the tuples with associated length t arrive at node v at time t. We omit the proof here, since it is similar to that of Lemma 1 in Section 4.

In order to retrieve the ℓ-MRP, $\overline{P}_{sd}^{\ell}$, that corresponds to a tuple (ℓ, R, v), follow the node v that corresponds to the third component of the tuple and choose the tuple $(\ell - \ell_{vd}, \cdot, \cdot)$ from $L(v)$. Since $L(v)$ keeps only the tuples with the highest sustainabilities for all possible lengths of paths from s to v, there exists only one tuple with associated length $\ell - \ell_{vd}$ in $L(v)$. We can retrieve the entire path by doing this process recursively until we get a null value at the third component of the tuple.

We now show an optimal subpath property for the MRP problem, which allows the use of dynamic programming. Let $P = (v_0, v_1, \ldots, v_k)$ be the ℓ-MRP from v_0 to v_k. Then the subpath $Q = (v_0, \ldots, v_i)$ of P is the ℓ_i-MRP from v_0 to v_i, where ℓ_i is the length of Q. Suppose for a contradiction, that there exists another path $Q' = (v_0, u_1, \ldots, u_j, v_i)$ with length ℓ_i and $Re(Q') > Re(Q)$. Then at time step ℓ_i, node v_i gets two tuples $q = (\ell_i, R, v_{i-1})$ and $q' = (\ell_i, R', u_j)$ corresponding to the paths Q and Q', respectively. Let the sustainability of path $(v_{i+1}, \ldots, v_k)$ with starting time ℓ_i be $\bar{R}$. Since we assumed that $Re(Q') > Re(Q)$, the path $P' = Q(v_{i+1}, \ldots, v_k)$ has sustainability $Re(Q') \cdot \bar{R} > Re(Q) \cdot \bar{R}$. This contradicts the assumption that P is an ℓ-MRP from v_0 to v_k.

The following theorem summarizes the properties of the MRP algorithm. We omit the full proof of this theorem due to space limitations.

Theorem 1. *The MRP algorithm finds the most reliable path for a unit-size message in $O(nM)$ time, where M is the upper bound of the number of time steps chosen.*

3.2 The MSP algorithm: constant-size messages

We generalize the algorithm of Section 3.1 to the case where the data size σ may be greater than one. In this case, the edge capacities play a concrete role in computing the path sustainabilities. There are two main differences between the algorithm in this section and the MRP algorithm. One is that when computing σ-MSPs at node v from a σ-MSP from s to a neighbor u of v, due to the nonuniform edge capacities, we may need to recompute the sustainability of the σ-MSP from s to u. The other difference is that the algorithm should be able to check for a cycle that causes a head blocking problem. This can be done by including the last σ nodes of a path in the tuples sent to neighboring nodes.

Before presenting the algorithm, we define the concept of *possible capacities*. Suppose there exists a path Q from node s to node v of capacity $c(Q)$. If $c \leq c(Q)$, then c is said to be a *possible capacity* of path Q. Furthermore, if there exists a path Q from s to v with length ℓ, we say that (c, ℓ) is a *possible capacity-length pair* from s to v, for any $c \leq c(Q)$.

We redefine the sustainability of an edge and the sustainability of a path in order to explicitly relate them to a specific possible capacity. The sustainability of edge (i, j) for the transmission of σ units of data for a possible capacity c of the transmission path is given by

$$r_{ij}^{\sigma,c}(t_i) = \prod_{t=t_i}^{\lceil \sigma/c \rceil + \ell - 1} r_{ij}(t), \tag{3}$$

where t_i denotes the starting time when edge (i, j) is used in the transmission. Equation(3) will be used instead of Equation(1) to account for possible capacities. A (σ, ℓ, c)-MSP from s to v is a (σ, ℓ)-MSP from s to v with possible capacity c. We change the notation P_{sv}^{ℓ} to $P_{sv}^{\ell,c}$ to allow for the possible capacity of a path. Equation(2), which gives the sustainability of a path $P_{v_0 v_k}^{\ell,c} = (v_0, v_1, \ldots, v_k)$ can be rewritten as

$$Re^{\sigma,c}(P_{v_0 v_k}^{\ell,c}) = \prod_{i=0}^{k-1} r_{v_i v_{i+1}}^{\sigma,c}(t_{v_i}),$$

where $\ell = \sum_{i=0}^{k-1} \ell_{v_i v_{i+1}}$ and c is a possible capacity of the path. We may omit the superscript c from $Re^{\sigma,c}$, whenever clear from the context.

Our algorithm keeps track of all possible capacity-length pairs (c, ℓ) in addition to all possible sequences of the last σ nodes on any path to the neighbors of a node v. We call these sequences the *σ-nearest subpaths* of v. If there exist two or more different paths with the same possible capacity-length pair and the same σ-nearest subpath, then our algorithm keeps the path with the highest sustainability. Each node v in the network G maintains a set of tuples of the form (c, ℓ, R, Q) where for some path P from s to v, c is a possible capacity of P, ℓ is the length of P, R is the corresponding sustainability of P, and Q is the σ-nearest subpath of v in P.

The MSP algorithm works just like the MRP algorithm, except for checking and detecting head blocking cycles. Note that the path found by our MSP algorithm may not be simple (it may contain cycles of length greater than σ).

Algorithm MSP(G, s, d)

1. *Initialization: during time step 0*:

1.1 The source node s initializes $R = Re(P_{ss}^{0,c}) = 1$ for all $c \leq \sigma$ and $L(s) = \{(c, 0, 1, null) | \forall c \leq \sigma\}$
1.2 Node s sends only the tuples $(c, \ell_{sv}, R \times r_{sv}^{\sigma;c}(t_s), (null, \ldots, null, s))$ with $c \leq \min(\sigma, c_{sv})$ to its neighbor, for all (s, v) in E.

2. *During time step* $t \leq M$:
   2.1 Each node v chooses the tuple $(c, t, R, (u_1, \ldots, u_\sigma))$ that has the maximum sustainability among all tuples of the form $(c, t, \cdot, (u_1, \ldots, u_\sigma))$ that v received during time step t.
   2.2 If $(u_i, u_{i+1}) \neq (u_\sigma, v)$, for all $i = 1, \cdots, \sigma - 1$, insert $(c, t, R, (u_1, \ldots, u_\sigma))$, for all c and for all $(u_1, \ldots, u_\sigma)$, into $L(v)$.
   2.3 If node v is not the destination, then for each tuple $(c, t, R, (u_1, \ldots, u_\sigma)) \in L(v)$, v sends to neighbor w tuples $(c, t + \ell_{vw}, R \times r_{vw}^{\sigma;c}(t_v), (u_2, \ldots, u_\sigma, v))$ with $c \leq c_{vw}$, for all (v, w) in E.
3. *Termination: time step* $M + 1$:
   3.1 Node d computes a MSP of length ℓ from s to d by choosing the tuple $q = (c, \ell, R, Q)$ from $L(d)$ that maximizes the sustainability R.

In the MSP algorithm, Step 2.2 ensures paths without cycles of length less than or equal to σ are considered. This is the basis for the proof of correctness of the MSP algorithm.

Theorem 2. *The MSP algorithm finds the most sustainable path for a message of size σ in $O(M\sigma\alpha n^{\sigma+1})$ time, where $\alpha = \min\{\sigma, \max_{(i,j)\in E}(c_{ij})\}$. In particular, if σ is a constant, the MSP algorithm runs in polynomial time on n.*

Proof. By the nature of the MSP algorithm, $L(d)$ will have exactly one tuple with associated length ℓ, for all possible capacities and for all possible σ-nearest subpaths of node d, at the end of time step M. Therefore the tuple computed in Step 3.1 is the σ-MSP from s to d. For the time bound of the algorithm, we observe that it takes constant time to calculate a tuple to be sent to the neighbors and to compare any two tuples. In Step 2.1, each node receives at most αn^σ tuples from an adjacent node. Thus it takes $O(\alpha n^{\sigma+1})$ time to choose the tuple with the maximum sustainability. In Step 2.2, for each selected tuple, only $\sigma - 1$ comparisons are necessary. Hence, Step 2.2 takes $O(\sigma\alpha n^{\sigma+1})$ time, and Step 2.3 takes $O(\alpha n^{\sigma+1})$ time. Therefore, the overall time complexity is $O(M\sigma\alpha n^{\sigma+1})$. If σ is a constant, this running time is polynomial on n (since M is also polynomial on n).

Note that the algorithm works correctly for any message size. However, if the size of the message is not a constant, we can no longer guarantee that it will run in time polynomial in n.

4 The MSRP Problem

In this section we present a polynomial time algorithm for finding most sustainable paths in circuit-switching networks, for arbitrary size messages.

In the circuit-switching routing model, once a path is established at the connection setup phase, it is reserved for the entire duration of the message transmission. Thus, the sustainability of a path is the product of the edge reliabilities of all edges in the path over all the time steps needed to transmit the message. We call the problem of finding a most sustainable path from s to d in circuit-switching networks *the most sustainable reserved path* (MSRP) problem. In particular, if the size of the message to be transmitted is σ, we call it the σ-MSRP problem.

The only difference from the network model of the previous section is the way the sustainability of an edge is defined. We keep the same notation as in the previous section unless specified. Before formalizing edge sustainability in this context, we define *the actual transmission duration* D of a path as the time between transmission of the first packet by the source and the receipt of the last packet by the destination. For each capacity-length pair (c, ℓ) at node v, the actual transmission duration D of a path from s to v of length ℓ and capacity c is $D = \lceil \sigma/c \rceil + \ell - 1$.

The sustainability of edge (i, j) for the entire duration of the transmission of a message of size σ, assuming that c is a possible capacity of a transmission path of length ℓ, is given by

$$R_{ij}^{\sigma,c} = \prod_{t=0}^{D} r_{ij}(t), \tag{4}$$

where D is the actual transmission duration of the path (which depends on ℓ). The sustainability of the path $P = (v_0, \ldots, v_k)$ is given by

$$Re^{\sigma,c}(P) = \prod_{i=0,\ldots,k-1} R_{v_i v_{i+1}}^{\sigma,c}.$$

Let P be a path whose actual transmission time is at most T. A (T, σ)-*most sustainable reserved path*, or for short (T, σ)-MSRP, for a message of size σ from s to d is a path P which maximizes $Re^{\sigma,c}(P)$. In this scenario, head blocking is no longer a concern since an MSRP is always simple, by the way the edge sustainabilities are defined in this model.

We now propose a polynomial time algorithm for the σ-MSRP problem. This algorithm uses the T-MSRP algorithm, which finds a (T, σ)-MSRP from s to d, as a subroutine. If path P is a (T, σ)-MSRP, where T is equal to the actual transmission duration of P, then P is a candidate for being a σ-MSRP. In other words, a σ-MSRP is the one with the highest path sustainability among all (T, σ)-MSRPs whose actual transmission durations are equal to T.

Our T-MSRP algorithm is similar in essence to the MSP algorithm. However, we introduce some pruning rules which help us reduce the number of subpaths we need to keep (these prunning rules do not apply to the MSP algorithm). At the beginning of the algorithm we do not know the actual transmission duration of the paths from s to t. Hence, we will choose different maximum transmission durations T arbitrarily, and then check whether there is a candidate path for being an MSRP, whose actual transmission duration is in fact equal to T.

The algorithm works as follows. Each node v will maintain a set of tuples $L(v)$ of the form (c, ℓ, R, T, u). Node v will update $L(v)$ upon learning the tuple

(c, ℓ, R, T, u) received from one of its neighbors u, where $c \leq \min\{\sigma, c_{uv}\}$. Node v will select the tuple which maximizes R among the tuples $(c, \ell, R, T, \cdot)$ received from its neighbors and will add it to $L(v)$. Node v will calculate $Re^{\sigma,c}(P)$, the sustainability for possible capacity c of the path P of length $\ell + \ell_{vw}$ composed of the path from s to v associated with the tuple $(c, \ell, R, T, \cdot)$ followed by edge (v, w), for all $(v, w) \in E$. Then, if $\lceil \sigma/c \rceil + \ell + \ell_{vw} - 1 \leq T$ for a possible capacity c, the sustainability of path P *as a subpath of a path from s to d* is given by

$$Re^{\sigma,c}(P) = R \cdot R^{\sigma,c}_{vw}(T), \tag{5}$$

where $R^{\sigma,c}_{vw}(T) = \prod_{t=0}^{T} r_{vw}(t)$. We use $R^{\sigma,c}_{ij}(T)$ to denote the sustainability of edge (i, j) for maximum transmission duration T, in order to distinguish it from Equation (4); we will also use the notation $R^{\sigma,c}_{P}(T)$ to denote the sustainability of path P for maximum transmission duration T. If node v does not have another tuple of the form $(c, \ell + \ell_{vw}, \cdot, T, \cdot)$ in $L(v)$, then v will add the tuple $(c, \ell + \ell_{vw}, Re^{\sigma,c}(P), T, v)$ to $L(v)$. Otherwise, v will keep the tuple with higher path sustainability and ignore the other tuples of the form $(c, \ell + \ell_{vw}, \cdot, T, \cdot)$.

In some cases, we may be able to prune a path from s to node v, if at some point we can guarantee that this path will not lead to a σ-MSRP from s to d. In the following paragraphs, we present three pruning rules which we use in our algorithm. Let $(c^*, t^*, R^*, T^*, \cdot)$ where $T^* = \lceil \sigma/c^* \rceil + t^* - 1$ is the tuple with the best sustainability found by our algorithm so far at node d. Suppose there exists a tuple $q = (c, t, R, T, u)$ at node v — associated with path Q from s to v — such that the actual transmission duration $D = \lceil \sigma/c \rceil + t - 1$ of Q is greater than T^* and the sustainability R is less than R^*. Since any path P from s to d which extends the path Q has longer transmission duration than that of Q, the final sustainability of path P will be less than R which is less than R^*. Thus the first pruning rule says that we can prune the path Q at node v — that means node v does not propagate the tuple calculated from q to its neighbors.

We can also prune path Q if $D > T$, since in this case, we cannot transmit the whole message in time T through this path, and so the information about the sustainability R is not well-defined. Thus the second pruning rule says that node v does not add the tuple q to $L(v)$ if $D > T$.

The third pruning rule compares paths from s to node $v \neq d$, with respect to the same total transmission duration. Suppose there are two paths $P = (s = u_1, u_2, \ldots, u_p = v)$ and $P' = (s = w_1, w_2, \ldots, w_{p'} = v)$ from s to node v associated with the tuples $q = (c, \ell, R, T, \cdot)$ and $q' = (c', \ell', R', T, \cdot)$, respectively. Assume that $R > R'$ and $c \geq c'$. Consider a path $Q = (v = v_1, v_2, \ldots, v_k = d)$ from v to d whose capacity is at least c. Then

$$\begin{aligned} R^{\sigma,c}_{P \cup Q}(T) &= \prod_{i=1}^{p-1} R^{\sigma,c}_{u_i u_{i+1}}(T) \times \prod_{j=1}^{k-1} R^{\sigma,c}_{v_j v_{j+1}}(T) \\ &> \prod_{i=1}^{p'-1} R^{\sigma,c'}_{w_i w_{i+1}}(T) \times \prod_{j=1}^{k-1} R^{\sigma,c'}_{v_j v_{j+1}}(T) = R^{\sigma,c'}_{P' \cup Q}(T). \end{aligned}$$

Since $Re^{\sigma,c}_{P\cup Q}(T)$ is strictly greater than $Re^{\sigma,c'}_{P'\cup Q}(T)$ for any choice of Q, intuitively, we might want to prune the path P' at node v. However if $\ell > \ell'$, it is still possible that P' will lead to a T-most sustainable path, whereas P will not. path from s to d. It may happen that the actual transmission duration of the path obtained by extending path P becomes greater than T, whereas that of the path obtained by extending path P' is still less than T. A similar scenario can happen when $R > R'$, $\ell \leq \ell'$ and $c < c'$. Hence we can safely prune the path P' only if $\ell \leq \ell'$, $c \geq c'$, and $R > R'$.

Before presenting the T-MSRP algorithm, we briefly describe our MSRP algorithm for a message of size σ. The algorithm will call the T-MSRP algorithm for all $1 \leq T \leq M$, and keep the tuple $(c, \ell, R, T, \cdot)$ returned by T-MSRP algorithm if the actual transmission duration $\lceil \sigma/c \rceil + \ell - 1$ is equal to T. At the end of the algorithm, we will have all the tuples that correspond to (T, σ)-MSRPs from s to d, for all T. Hence the σ-MSRP from s to d will be obtained by choosing the tuple with maximum sustainability.

Now we present our algorithm T-MSRP, which computes a T-most sustainable path from s to d for a message of size σ. The parameters c^*, R^*, and T^* are obtained from the tuple $(c^*, \ell, R^*, T^*, \cdot)$ with maximum path sustainability where $\lceil \sigma/c^* \rceil + \ell - 1 = T^*$ that the MSRP algorithm found so far at node d.

Algorithm T-MSRP(G, s, d, σ,T, c^*, T^*, R^*)

1. *During time step* $t = 0$:
   1.1 The source node s initializes $R = Re^{\sigma,c}(P^0_{ss}) = 1$, for all $c \leq \sigma$, and $L(s) = \{(c, 0, R, T, null) | \, \forall c \leq \sigma\}$.
   1.2 For each neighbor v, node s sends to v only the tuples $(c, \ell_{sv}, R \times R^{\sigma,c}_{sv}(T), T, s)$, such that either $\lceil \sigma/c \rceil + \ell_{sv} - 1 \leq T^*$ or $R \geq R^*$, and $\lceil \sigma/c \rceil + \ell_{sv} - 1 \leq T$, where $c \leq \min\{\sigma, c_{sv}\}$. (by the first and second pruning rules)
2. *During time step* $t \leq T$:
   2.1 Each node v compares all the tuples of the form $(c, t, \cdot, T, \cdot)$ received from its neighbors at time step t.
   2.2 For each possible capacity c at node v,
      2.2.1 Choose the tuple $(c, t, R, T, \cdot)$ that has maximum sustainability among all tuples of the form $(c, t, \cdot, T, \cdot)$ v received.
      2.2.2 Node v compares $(c, t, R, T, \cdot)$ to each tuple $(c', t', R', T, \cdot)$ in $L(v)$.
         2.2.2.1 If there is a tuple $(c', t', R', T, \cdot)$ with $c \geq c'$, $t < t'$, and $R > R'$ then remove the tuple from $L(v)$. (by the third pruning rule)
         2.2.2.2 If there does not exist any tuple $(c', t', R', T, \cdot)$ with $c \leq c'$, $t > t'$, and $R < R'$ then insert $(c, t, R, T, \cdot)$ into $L(v)$. Otherwise, discard the tuple $(c', t', R', T, \cdot)$. (by the third pruning rule)
      2.2.3 If node v is not the destination and $(c, t, R, T, \cdot)$ is not discarded, then v computes $\bar{t} = t + \ell_{vw}$ and $\bar{R} = R \times R^{\sigma,c}_{vw}(T)$ and update the previous node to v, for all (v, w) in E.
      2.2.4 Node v sends to neighbor w tuples $(c, \bar{t}, \bar{R}, T, \cdot)$, such that either $\lceil \sigma/c \rceil + \bar{t} - 1 \leq T^*$ or $R \geq R^*$, and $\lceil \sigma/c \rceil + \bar{t} - 1 \leq T$, where $c \leq \min\{\sigma, c_{vw}\}$. (by the first and second pruning rules)

3. *During time step $t = T + 1$:*
   3.1 Node d computes a (T, σ)-MSRP from s to d by choosing the tuple $q = (c, t, R, T, v)$ that maximizes the sustainability R.
   3.2 Return the tuple q.

We now present three lemmas that form the foundation for the proof of correctness of our MSRP algorithm.

Lemma 1. *At time step t, a node v only receives tuples with associated length t from its neighbors.*

Proof. We prove the lemma by induction on time step t. When $t = 0$, no node in the network receives any tuple. Assume the above lemma is true at any time $t' < t$. We prove the lemma also holds for time step t.

At time step $t - \ell_{vw}$, node v received all tuples of the form $(c, t - \ell_{vw}, \cdot, T, \cdot)$, by the induction hypothesis. After selecting the tuple $(c, t - \ell_{vw}, R, T, \cdot)$ which has the highest sustainability among tuples of the form $(c, t - \ell_{vw}, \cdot, T, \cdot)$, v sends $q = (c, t - \ell_{vw} + \ell_{vw}, \cdot, T, v)$ to its neighbor w still at time step $t - \ell_{vw}$. Since v does not receive any other tuples of the form $(c, t - \ell_{vw}, \cdot, T, \cdot)$ at any time step other than $t - \ell_{vw}$, q is the only tuple sent by node v to node w with associated capacity-length pair (c, t). Since it takes ℓ_{vw} time steps for all tuples bound from v to w to traverse edge (v, w) (there are at most as many possible capacities — and thus tuples — as the capacity of edge (v, w); therefore we assume without loss of generality that all tuples in consideration can be sent from v to w at time $t - l_{vw}$) all the tuples that left v to w at time step $t - \ell_{vw}$ must reach node w at time t. In particular, q reaches node w at time t. Since this is true for any pair of adjacent nodes v and w and any possible capacity, the lemma holds.

Lemma 2. *A T-most sustainable reserved path obtained by the T-MSRP algorithm does not contain a cycle.*

Proof. Suppose a path P from s to d contains a cycle C. Then we can construct a path P' from s to d by removing the edges in C from P, leaving s and d still connected by the edges in P'. Thus the number of edges in P' is less than the number of edges in P and P' is a subpath of P. Then $R_P^{\sigma,c}(T) = R_{P'}^{\sigma,c}(T) \times R_C^{\sigma,c}(T)$ for any possible capacity c. Since $0 < r_{ij}(t) < 1$, for all (i, j) in E, $0 < R_C^{\sigma,c}(T) < 1$. Hence $R_P^{\sigma,c}(T) < R_{P'}^{\sigma,c}(T)$

Lemma 3. *Algorithm T-MSRP finds a T-most sustainable reserved path for a message of size σ from s to d.*

Proof. The tuples that a node v receives from a neighbor u at time step t have length t. For each possible capacity c, node v chooses the tuple $q = (c, t, R, T, \cdot)$ that has the maximum sustainability R among all the tuples of the form $(c, t, R, T, \cdot)$ that v received. Then v computes a new tuple $\bar{q} = (c, t + \ell_{vw}, R \times R_{vw}^{\sigma,c}(T), T, \cdot)$, for each $(v, w) \in E$ based on the tuple q, and sends $\bar{q}$ to its neighbor w if the tuple does not satisfy any one of the pruning conditions. By Lemma 1, q is the only tuple kept in $L(v)$ at time step t with associated capacity-length pair $(c, t+\ell_{vw})$. Thus, at time step $T+1$, node d will have only one tuple in $L(d)$ for each capacity-length pair (c, ℓ) if there exists a path from s to d with length ℓ and a possible capacity c. The tuple with the maximum sustainability among the tuples in $L(d)$ represents the T-most sustainable path from s to d for a message of size σ and transmission duration T.

Since an MSRP has to be simple, it suffices to let the upper bound on the maximum path length, M, be equal to the number of edges in a longest path multiplied by the largest edge delay in the network. Since the delay of an edge is bounded by a polynomial on n, M is also bounded by a polynomial on n. For most multimedia applications, in particular for real-time ones, the end-to-end delay is one of the most important QoS requirements. Thus one may want to choose M to be the deadline that guarantees the desired timely delivery of messages a the upper bound on the maximum allowed path length.

Finally, we present an alorithm for computing a σ-MSRP of G.

Algorithm MSRP(G, s, d, σ)

1 Initialize $c^* = 1$, $R^* = 0$, and $T^* = 0$
2 For each possible transmission duration T $(1 \le T \le M)$,
  2.1 Call T-MSRP$(G, s, d, \sigma, T, c^*, T^*, R^*)$ and let $q = (c, \ell, R, T, u)$ be the tuple returned by T-MSRP.
  2.2 If $\lceil \sigma/c \rceil + \ell - 1 = T$, then keep the tuple q.
  2.3 If $R^* \le R$, then let $c^* = c$, $R^* = R$, and $T^* = T$.
3 The σ-MSRP is the path that corresponds to the tuple $(c^*, T^*, R^*, T^*, \cdot)$.

We can retrieve the entire path which is selected by MSRP algorithm as we did in Section 3.1.

Theorem 3. *The MSRP algorithm finds the most sustainable reserved path for a message of size σ in $O(\alpha^2 M^3)$ time, where $\alpha = \min\{\sigma, \max_{(i,j)\in E}(c_{ij})\}$.*

Proof. The correctness of the algorithm follows directly from the previous lemmas. We then show the bound on the running time. In the T-MSRP algorithm, Step 2.2 will be executed at most α times. Since at time step t, node v has at most αM tuples in $L(v)$, the comparisons in Step 2.2.2 takes $O(\alpha M)$ time. The computations in Step 2.2.3 takes $O(n)$ time in the worst case. Hence the T-MSRP algorithm takes $O(\alpha M(\alpha M + n)) = O(\alpha^2 M^2)$ time (we can assume that $M \ge n$, without loss of generality). Therefore, the time complexity of the MSRP algorithm is $O(\alpha^2 M^3)$.

5 Conclusion

In this paper, we formalized the problem of finding most sustainable paths in networks with time-dependent edge reliabilities. We considered two network models: the wormhole and circuit-switching models. For the former model, we presented a fully-distributed polynomial time algorithm for selecting a MSP for the transmission of constant-size messages. Even though it works correctly for any message size, due to the size of tuples that are exchanged between nodes during the execution of the algorithm, we can no longer guarantee that it will run in polynomial time on n if the sizes of messages are not constant. For the latter model, we presented the MSRP algorithm, which finds a σ-MSRP from the source to the destination in polynomial time, where σ is an arbitrary polynomial on n.

One natural extension of the work presented in this paper would be to seek for polynomial time algorithms that compute a σ-MSP in the wormhole routing model for an arbitrary message size σ, with time-dependent edge reliabilities. This problem may prove to be very hard to solve. Thus, approximation algorithms for this problem would also be of great interest.

References

1. Ahuja, R. K.: Minimum cost reliability problem. Computers and Operations Research 16 (1988) 83-89
2. Barrett, C., Bisset, K., Konjevod, G., Jacob, R., Marathe, M.: Routing problems in time-dependent and labeled networks, submitted to SODA 2002
3. Bertsekas, D., Gallager, R.: Data Networks. Prentice Hall, (1992)
4. Chen, Y. L., Chin, Y. H.: The quickest path problem. Computers and Operations Research 17, no. 2 (1990) 153-161
5. Cooke, K., Halsey, E.: The shortest route through a network with time dependent internodal transit times. Journal of mathematical analysis and applications 14 (1966) 493-498
6. Gibson, G. A., Vitter, J. S., Wilkes, J.: Strategic Direction in Storage I/O issues in large-scale computing. In Proceedings of the ACM Workshop on Strategic Directions in Computing Research, ACM Computing Surveys 28, no. 4 (1996) 779-793
7. Handel, R., Huber, M. N.: Integrated Broadband Networks: An Introduction to ATM-Based Networks. Addison-Wesley (1991)
8. Inktomi Inc. Streaming media caching white paper. Inktomi Corporation, Technical Report (1999)
9. Miller-Hooks, E., Patterson S. S.: The time-dependent quickest flow problem. Submitted to Operations Research 2000
10. McDysan, D. E., Spohn, D. L.: ATM: Theory and Application. McGraw-Hill (1995)
11. Orda, A., Rom, R.: Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. Journal of the Association for Computing 37, no. 3 (1990) 607-625
12. Orda, A., Rom, R.: Distributed shortest-path protocols for time-dependent networks. Distributed Computing 10 (1996) 49-62
13. Scheideler, C.: Universal routing strategies for interconnection networks. Springer (1998)
14. Sung, K., Bell, M., Seong, M., Park, S.: Shortest paths in a network with time-dependent flow speeds. European Journal of Operational Research 121 (2000) 32-39
15. Tragoudas, S.: The most reliable path transmission. Performance, Computing, and Communications Conference, 1999 IEEE International (1999) 15-19
16. Vogel, R., Herrtwich, R.G., Kalfa, W., Wittig, H., Wolf, L.C.: QoS-based routing of multimedia streams in computer networks. IEEE Journal on Selected Areas in Communications 14, no. 7 (1996) 1235-1244
17. Wang, Z., Crowcroft, J.: Quality-of-service routing for supporting multimedia applications. IEEE Jornal on Selected Areas in Communications 14, no. 7 (1996) 1228-1234