

Multi-variable cost function for Application Layer Multicast routing

Tien Anh Le, Hang Nguyen, Hongguang Zhang

Abstract—Cost function is an essential part in Application Layer Multicast (ALM) routing algorithms. It is from a cost function that we can calculate links' costs and then build the data delivery tree for multicasting. Unfortunately, cost function remains an almost untouched research area in ALM routing. In this research, we propose a new multi-variable cost function considering various end-to-end QoS parameters simultaneously. The mathematical derivation process is also described in details so that one can apply it to obtain other multi-variable cost functions according to their specific requirements. The newly proposed multi-variable cost function can avoid congestion before it happens, preventing the data delivery tree from being frequently or unnecessarily changed while still be adaptable to the dynamic requirements of different applications. With theoretical analysis, we have proved that the new cost function can provide better performance for ALM routing algorithms compared to conventional cost functions.

Index Terms—application layer multicast routing; cost function; resource allocation; traffic control; end-to-end QoS routing;

I. INTRODUCTION

The Internet was originally built for unicast or one-to-one applications. Nowadays, it has to serve a large number of multimedia services such as multimedia conference. These types of multicast services put a big load on the unicast infrastructure of the Internet. IP-Multicast[1] is the first attempt to solve this problem. It is so far the most efficient multicast mechanism to deliver the data over each link of the network only once. However, many deployment problems are still preventing IP-Multicast from being supported worldwide[2]. An alternative solution is Application Level Multicast(ALM). The key concept of ALM is the implementation of multicasting functionality as an application service instead of a network service. It has excellent advantages over IP-Multicast: easier and possibly immediate deployment over the Internet without any modification of the current infrastructure and adaptable to a specific application. In a tree-push ALM, a data distribution tree is built first, then the data is actively distributed from the source node to intermediate peers until reaching all peers in the multicast tree[3]. In order to build an ALM distribution tree, we must have costs of all available end-to-end links. Those costs can only be calculated by using a cost function.

The main contribution of this research is to design a new multi-variable cost function of the end-to-end delay and bandwidth taking into account advantages of application layer links. We also propose in details the mathematical derivation process of the new multi-variable cost function and analyze its performance. This analysis and derivation process can be further applied to find other forms of cost functions with other QoS parameters. A theoretical anal-

ysis will also be performed to compare the newly found cost function with conventional ones. The new cost function is capable of preventing congestion on each link before it occurs by assigning higher costs to nearly-overloaded end-to-end links. The new cost function also considers the dynamic characteristics of end-to-end links of the overlay environment.

The rest of the paper is organized as follows. Section II provides a survey on the state-of-the-art cost functions and analyzes their limitations. The proposed multi-variable cost function will be described and derived in section III. By theoretical analysis, we will compare the new multi-variable cost function with conventional ones to show its advantages in section IV. Finally, we conclude and give out some possible future works in Section V.

II. CONVENTIONAL COST FUNCTIONS

Conventional cost functions are either empirical or heuristic. Among all available cost functions for ALM routing that we have found, none of them has a mathematical derivation nor a clear citation. In most of the ALM routing algorithms, the state of the network, on which the routing algorithm is presented, readily associates some costs with each link. Thus they do not address how the link cost function should be defined so as to efficiently distribute allocated resources over the network[4]. Also in[4], several kinds of cost functions have been investigated.

$$LinkCost = \frac{RsvBw + ReqBw}{LinkCap} \quad (1)$$

In which:

- RsvBw: The amount of bandwidth currently in use by existing connections,
- ReqBw: The amount of bandwidth requested by the newly arriving group,
- LinkCap: Total bandwidth of the link.

The main idea of using the cost function (1) is to choose a tree that is least-loaded and at the same time, to minimize the total amount of bandwidth to be consumed by the new connection.

$$LinkCost = 1 + \frac{RsvBw}{LinkCap} \quad (2)$$

The main idea of using the cost function of (2) (a variation of (1)) is that hop count, a static link metric, plays a dominant role in the link cost.

$$LinkCost = \frac{1}{LinkCap - (RsvBw + ReqBw)} \quad (3)$$

In (3), the cost increases exponentially with the utilization of the link. According to[4], (3) appears more attractive

than (1) and (2) because it can better distribute the load over the network by avoiding the use of highly loaded links, and thus links don't likely become saturated for future connections. Furthermore, it gives preference to shorter paths (with less number of link) as long as links are not heavily loaded.

In[5], the cost function (4) was used because it has many desirable practical characteristics. It decreases with the delay, it is convex, it assigns infinitely high cost when the required delay guarantee approaches zero, and a fixed minimal link usage cost C^l , even if no guarantee is required. The constant θ determines how fast the cost grows for low delays and the constant A^l is used as a scaling constant.

$$C^l(d) = \frac{A^l}{d^\theta} \quad (4)$$

In[6], the cost function on each link is calculated by (5)

$$c_i(x) = \begin{cases} \frac{\kappa_i}{x - \kappa_i} & , \text{ if } x > \kappa_i \\ \infty & , \text{ if } x \leq \kappa_i \end{cases} \quad (5)$$

In which:

- κ_i : Minimal delay that can be guaranteed on link i when utilizing all of its available resources,
- x_i : Requested delay of the new connection.

Although delay is used as a sample, any other QoS parameter such as bandwidth, jitter, packet loss can be used. However, the heuristics function (5) is only single-variable, therefore it cannot consider other parameters simultaneously.

Another cost function (6) considering several QoS parameters has been used in[7], again, without any mathematical proof.

$$C(\beta, \delta, \psi) = \delta + \frac{K_1}{\beta} + K_2 \cdot e^{\frac{\kappa_3}{\psi}} \quad (6)$$

In which:

- β : Residual bandwidth,
- δ : Residual buffer space,
- ψ : Estimated delay bound.

The scaling factor K_i allows us to modulate the relationship between β, δ, ψ even further, although it is still unclear how bandwidth, buffer, and delay units could be added exactly together[7],[8].

In [9], a single-variable cost function has been derived and its performance has been simulated under real conditions. The results have shown that, the single-variable cost function can only build a good multicast tree in certain conditions. A multi-variable cost function is highly demanded to build a more reliable multicast tree. From the best of our knowledge, conventional cost functions are mainly applied in network-layer routing. Our work has been one of the pioneers who have made full investigations on application layer multicast routing considering both network conditions and application's requirements.

III. PROPOSED MULTI-VARIABLE COST FUNCTION

QoS parameters can be either bandwidth-type (meaning that the requested bandwidth is always smaller than or

equal to the maximum available bandwidth) or delay-type (meaning that the requested delay is always greater than or equal to the minimum available delay).

A. Problem formation

Problem: Find a multi-variable cost function which can simultaneously consider varied bandwidth and delay requests from the application and maximum guaranteed resources from the underlay network. The cost function must be able to assign increasingly higher costs for nearly-saturated end-to-end links to prevent congestion.

B. Lemma 1: Bandwidth-type cost function

Assume we have on the end-to-end link i : A total available bandwidth of κ_w , and a requested bandwidth of x_w , we must find the bandwidth-type cost function: $f(x_w)$. Since κ_w is the maximum available bandwidth when using all available resources on link i , so $0 \leq x_w \leq \kappa_w$. With time, according to the application's requirements, x_w may be varied by an amount of Δx_w causing the cost to have the current value of $f(x_w + \Delta x_w)$, so this current value of the cost function depends on:

- The previous cost: $f(x_w)$,
- The increment of cost which is proportional to:
 - The previous cost: $f(x_w)$,
 - The ratio between the increment of requested bandwidth and the total requested bandwidth: $\frac{\Delta x_w}{x_w + \Delta x_w}$,
- The decrement of cost which is proportional to:
 - The ratio between the decrement of the remaining available bandwidth and the maximum available bandwidth $\frac{(\kappa_w - x_w - \Delta x_w)}{\kappa_w}$.

Finally, we have:

$$f(x_w + \Delta x_w) = f(x_w) \cdot \left[1 + \frac{\frac{\Delta x_w}{x_w + \Delta x_w}}{\frac{(\kappa_w - x_w - \Delta x_w)}{\kappa_w}} \right] \quad (7)$$

From (7) we have:

$$\begin{aligned} & \lim_{\Delta x_w \rightarrow 0} \frac{f(x_w + \Delta x_w) - f(x_w)}{\Delta x_w} \\ &= \lim_{\Delta x_w \rightarrow 0} f(x_w) \frac{\frac{\kappa_w}{(x_w + \Delta x_w)} \cdot \frac{1}{(\kappa_w - x_w - \Delta x_w)}}{\frac{\kappa_w}{x_w(\kappa_w - x_w)}} \\ &\Leftrightarrow f'(x_w) = f(x_w) \cdot \frac{\kappa_w}{x_w(\kappa_w - x_w)} \end{aligned} \quad (8)$$

Replacing $f(x_w)$ by y and $f'(x_w)$ by $\frac{dy}{dx_w}$; from (8) we have an ordinary differential equation:

$$\frac{dy}{dx_w} = y \frac{\kappa_w}{x_w(\kappa_w - x_w)} \quad (9)$$

Solve the ordinary differential equation (9), we find the bandwidth-type cost function:

$$\begin{aligned} & \frac{dy}{y} = \frac{\kappa_w}{x_w(\kappa_w - x_w)} dx_w \\ &\Leftrightarrow \int \frac{dy}{y} = \int \frac{\kappa_w}{x_w(\kappa_w - x_w)} dx_w \end{aligned}$$

$$\int \frac{x_w + (\kappa_w - x_w)}{x_w(\kappa_w - x_w)} dx_w = - \int \frac{d(\kappa_w - x_w)}{\kappa_w - x_w} + \int \frac{dx_w}{x_w}$$

$$\Leftrightarrow \ln(y) = \ln(x_w) - \ln(\kappa_w - x_w) + c \Leftrightarrow y = \frac{\Phi \cdot x_w}{(\kappa_w - x_w)} \quad (10)$$

C. Lemma 2: Delay-type cost function

We can see that, the required delay parameter (x_d) has a reversed characteristic against the required bandwidth parameter (x_w). So by replacing $\dot{x}_d = \frac{1}{x_d}$, $\dot{\kappa}_d = \frac{1}{\kappa_d}$, and $d\dot{x}_d = d(\frac{1}{x_d}) = -\frac{dx_d}{x_d^2}$ into (9) we have:

$$\frac{dy}{d\dot{x}_d} = y \frac{\dot{\kappa}_d}{\dot{x}_d(\dot{\kappa}_d - \dot{x}_d)} \Leftrightarrow -\frac{dy}{dx_d} = y \frac{\frac{1}{\kappa_d}}{\frac{1}{x_d}(\frac{1}{\kappa_d} - \frac{1}{x_d})}$$

$$\Leftrightarrow \frac{dy}{dx_d} = y \frac{1}{\kappa_d - x_d} \quad (11)$$

Equation (11) is the ordinary differential equation to derive the delay-type cost function. From (11), we have:

$$\frac{dy}{y} = \frac{dx_d}{\kappa_d - x_d} \Leftrightarrow \int \frac{dy}{y} = - \int \frac{d(x_d - \kappa_d)}{x_d - \kappa_d}$$

$$\ln(y) = -\ln(x_d - \kappa_d) + \ln(c) \Leftrightarrow y = \frac{c}{x_d - \kappa_d}$$

$$\Leftrightarrow y = \frac{\Psi \cdot \kappa_d}{x_d - \kappa_d} \quad (12)$$

D. Derivation of the multi-variable cost function

We now try to derive the **bandwidth-delay cost function** $u(x_w, x_d)$ considering two independent QoS parameters: bandwidth (x_w) and delay (x_d) at the same time. From (9) and (11), we have:

$$\frac{x_w(\kappa_w - x_w)}{\kappa_w} u_{x_w} + (\kappa_d - x_d) u_{x_d} = u \quad (13)$$

In which $u_{x_w} = \frac{\partial u}{\partial x_w}$, and $u_{x_d} = \frac{\partial u}{\partial x_d}$.

Equation (13) is a *quasi linear first order partial differential equation*, we will solve it to obtain our bandwidth-delay cost function.

Let $x_w = x_w(s)$, $x_d = x_d(s)$, $u = u(x_w(s), x_d(s))$, then:

$$\frac{\partial x_w}{\partial s} \cdot u_{x_w} + \frac{\partial x_d}{\partial s} \cdot u_{x_d} = \frac{\partial u}{\partial s} \quad (14)$$

Compare (13) and (14), we have:

$$\left\{ \begin{array}{l} \frac{\partial x_w}{\partial s} = \frac{x_w(\kappa_w - x_w)}{\kappa_w} \\ \frac{\partial x_d}{\partial s} = \kappa_d - x_d \\ \frac{\partial u}{\partial s} = u \end{array} \right. \quad (15)$$

A *constant of integration* is obtained by eliminating s from two or more equations and integrating out. Such integration generates an arbitrary integration constant, which

may be viewed as a function of all the variables, but it is constant with respect to s . Let $\phi(x_w, x_d, u)$ be a constant of integration, since it is constant with respect to s , we can write:

$$\frac{d\phi}{ds} = 0 \Leftrightarrow \frac{\partial \phi}{\partial x_w} \cdot \frac{\partial x_w}{\partial s} + \frac{\partial \phi}{\partial x_d} \cdot \frac{\partial x_d}{\partial s} + \frac{\partial \phi}{\partial u} \cdot \frac{\partial u}{\partial s} = 0$$

$$\Leftrightarrow \frac{\partial \phi}{\partial x_w} \cdot \frac{x_w(\kappa_w - x_w)}{\kappa_w} + \frac{\partial \phi}{\partial x_d} \cdot (\kappa_d - x_d) + \frac{\partial \phi}{\partial u} \cdot u = 0 \quad (16)$$

Equation (16) is the *orthogonality property* of the vector (x_w, x_d) , we can use it to check whether ϕ has been obtained correctly.

In order to solve (13) we have to find two constants of integration from (15).

D.1 Finding the first constant of integration:

From (15), we have:

$$\frac{dx_w}{ds} = \frac{x_w(\kappa_w - x_w)}{\kappa_w} \Leftrightarrow \frac{du}{u} = \frac{\kappa_w \cdot dx_w}{x_w(\kappa_w - x_w)} \quad (17)$$

Since (17) and (9) have an identical form, we can use Lemma 1 to achieve the first constant of integration (18):

$$u = \frac{\Phi \cdot x_w}{(\kappa_w - x_w)} \Leftrightarrow \Phi = \frac{(\kappa_w - x_w)u}{x_w} \quad (18)$$

We now check the orthogonality property of the first constant of integration (18) by confirming (16):

$$\frac{\partial \Phi}{\partial x_w} \cdot \frac{x_w(\kappa_w - x_w)}{\kappa_w} + \frac{\partial \Phi}{\partial x_d} \cdot (\kappa_d - x_d) + \frac{\partial \Phi}{\partial u} \cdot u$$

$$= \frac{\partial \left(\frac{(\kappa_w - x_w)u}{x_w} \right)}{\partial x_w} \cdot \frac{x_w(\kappa_w - x_w)}{\kappa_w} + \dots$$

$$\dots + \frac{\partial \left(\frac{(\kappa_w - x_w)u}{x_w} \right)}{\partial x_d} \cdot (\kappa_d - x_d) + \frac{\partial \left(\frac{(\kappa_w - x_w)u}{x_w} \right)}{\partial u} \cdot u$$

$$= \frac{-\kappa_w \cdot u}{x_w^2} \cdot \frac{x_w(\kappa_w - x_w)}{\kappa_w} + 0 \cdot (\kappa_d - x_d) + \dots$$

$$\dots + \frac{(\kappa_w - x_w)}{x_w} \cdot u = 0 \quad (19)$$

By (19) and (16), we can confirm that the first constant of integration has been correctly found.

D.2 Finding the second constant of integration:

Similarly, using (15) and (12), we can find the second constant of integration having the form of:

$$\Psi = \frac{(x_d - \kappa_d)u}{\kappa_d} \quad (20)$$

We now check the orthogonality property of the first constant of integration (20) by confirming (16):

$$\frac{\partial \Psi}{\partial x_w} \cdot \frac{x_w(\kappa_w - x_w)}{\kappa_w} + \frac{\partial \Psi}{\partial x_d} \cdot (\kappa_d - x_d) + \frac{\partial \Psi}{\partial u} \cdot u$$

$$\begin{aligned}
&= \frac{\partial\left(\frac{(x_d - \kappa_d)u}{\kappa_d}\right)}{\partial x_w} \cdot \frac{x_w(\kappa_w - x_w)}{\kappa_w} + \dots \\
&\dots + \frac{\partial\left(\frac{(x_d - \kappa_d)u}{\kappa_d}\right)}{\partial x_d} (\kappa_d - x_d) + \frac{\partial\left(\frac{(x_d - \kappa_d)u}{\kappa_d}\right)}{\partial u} u \\
&= 0 \cdot \frac{x_w(\kappa_w - x_w)}{\kappa_w} + \frac{u}{\kappa_d} (\kappa_d - x_d) + \dots \\
&\dots + \frac{(x_d - \kappa_d)}{\kappa_d} \cdot u = 0 \quad (21)
\end{aligned}$$

By (21) and (16), we can confirm that the second constant of integration has been correctly found.

D.3 The general solution:

The equation $\Phi(x_w, x_d, u) = \text{constant}$, describes a relationship among x_w, x_d, u such as shown in (15). Notice that if $\Phi(x_w, x_d, u)$ is a constant, then $\mathcal{G}(\Phi)$ is also a constant ($\mathcal{G}(\cdot)$ is any arbitrary function). Similarly, if $\Psi(x_w, x_d, u)$ is a constant, then $\mathcal{H}(\Psi)$ is also a constant ($\mathcal{H}(\cdot)$ is any arbitrary function). We can set these two constants equal so that:

$$\mathcal{G}(\Phi) = \mathcal{H}(\Psi) \quad (22)$$

This provides a more general expression in x_w, x_d, u that solves the partial differential equation (13). The two arbitrary functions in (22) may be merged into one by letting $\mathcal{F}(\cdot) = \mathcal{G}^{-1}(\mathcal{H}(\cdot))$, then:

$$\Phi = \mathcal{F}(\Psi) \quad (23)$$

(23) is the general solution of the partial differential equation (13).

D.4 Checking the general solution:

From (18), (20), (23), we have:

$$\frac{(\kappa_w - x_w)u}{x_w} = \mathcal{F}\left(\frac{(x_d - \kappa_d)u}{\kappa_d}\right) \quad (24)$$

We will now check whether (24) is indeed a solution, and that the function \mathcal{F} can be completely general. Let u be expressed as a function of x_w and x_d , where x_w and x_d are still independent.

We take the total derivative of (24) to x_w and solve for u_{x_w} . Denoting $\mathcal{F}\left(\frac{(x_d - \kappa_d)u}{\kappa_d}\right) = \mathcal{F}(x_d, u)$, we have:

$$\begin{aligned}
&\frac{\kappa_w - x_w}{x_w} u_{x_w} - \frac{\kappa_w}{x_w^2} \cdot u = \frac{x_d - \kappa_d}{\kappa_d} \cdot u_{x_w} \cdot \mathcal{F}'(x_d, u) \\
\Leftrightarrow u_{x_w} &= \frac{\kappa_w \cdot \kappa_d \cdot u}{x_w [(\kappa_w - x_w)\kappa_d - x_w(x_d - \kappa_d)\mathcal{F}'(x_d, u)]} \quad (25)
\end{aligned}$$

Similarly, taking the total derivative of (24) to x_d and solve for u_{x_d} , we have:

$$\begin{aligned}
&\frac{(\kappa_w - x_w)u_{x_d}}{x_w} = \mathcal{F}'(x_d, u) \cdot \left[\frac{u + (x_d - \kappa_d)u_{x_d}}{\kappa_d} \right] \\
\Leftrightarrow u_{x_d} &= \frac{\mathcal{F}'(x_d, u) \cdot u \cdot x_w}{(\kappa_w - x_w)\kappa_d - \mathcal{F}'(x_d, u) \cdot (x_d - \kappa_d) \cdot x_w} \quad (26)
\end{aligned}$$

Replacing (25) and (26) into the left-hand side of (13) we have:

$$\begin{aligned}
&\frac{x_w(\kappa_w - x_w)}{\kappa_w} \cdot \frac{\kappa_w \cdot \kappa_d \cdot u}{x_w \cdot [(\kappa_w - x_w)\kappa_d - x_w(x_d - \kappa_d)\mathcal{F}'(x_d, u)]} + \\
&\dots + (\kappa_d - x_d) \cdot \frac{\mathcal{F}'(x_d, u) \cdot u \cdot x_w}{(\kappa_w - x_w)\kappa_d - \mathcal{F}'(x_d, u) \cdot (x_d - \kappa_d) \cdot x_w} \\
&= \frac{(\kappa_w - x_w) \cdot \kappa_d - (x_d - \kappa_d) \cdot \mathcal{F}'(x_d, u) \cdot x_w}{(\kappa_w - x_w)\kappa_d - x_w(x_d - \kappa_d) \cdot \mathcal{F}'(x_d, u)} \cdot u = u
\end{aligned}$$

We can obtain the right-hand side of (13), so (24) is exactly the general solution of the partial differential equation (13).

D.5 Fitting boundary conditions to the general solution:

Equation (24) provides us a general solution comprising of a family of arbitrary functions. We need to fix to a certain bandwidth-delay cost function by assigning boundary conditions to this general solution. From the natural characteristics of two independent QoS parameters: *requested bandwidth* (x_w) and *requested delay* (x_d), and their partial cost functions (10) and (12), we have these boundary conditions:

$$\begin{cases} \frac{x_w}{\kappa_w - x_w} = t^3 \\ \frac{x_d - \kappa_d}{u} = t^2 \end{cases} \quad (27)$$

Replacing (27) into (24) we have:

$$\mathcal{F}\left(\frac{t^2}{t}\right) = \frac{t^2}{t^3} \Leftrightarrow \mathcal{F}(t) = \frac{1}{t} \quad (28)$$

From (28) and (24), we have:

$$\mathcal{F}\left(\frac{(x_d - \kappa_d) \cdot u}{\kappa_d}\right) = \frac{\kappa_d}{(x_d - \kappa_d) \cdot u} = \frac{(\kappa_w - x_w)u}{x_w} \quad (29)$$

The specific solution of (13) is therefore:

$$u(x_w, x_d) = \sqrt{\frac{x_w}{\kappa_w - x_w} \cdot \frac{\kappa_d}{x_d - \kappa_d}} \quad (30)$$

Recursively, we can see that, the specific multi-variable cost function equals to the average multiplication of all partial cost functions $f_i(x_i)$:

$$u(x_1, x_2, \dots, x_n) = \sqrt[n]{\prod_{i=1}^n f_i(x_i)} \quad (31)$$

In general, we can build a cost function for as many variables as possible. However, while a multi-variable cost function can consider many QoS parameters at the same time, it should be noted that the multi-variable cost function does not always give a better result than the single-variable cost function. For example, the cost function with bandwidth, delay, and packet-loss can build a better multicast tree if many peers are using wireless access network with a high packet loss rate to join the multicast tree but when most of the peers are using a wired access network with a low packet loss rate, then that three-variable cost function may build a worse multicast tree than the two-variable cost function of only bandwidth and delay.

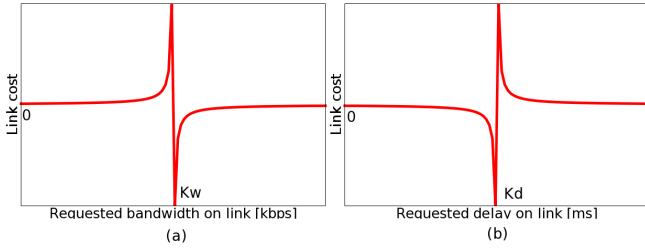


Fig. 1. Bandwidth-type (a) and delay-type (b) cost functions according to (10) and (12).

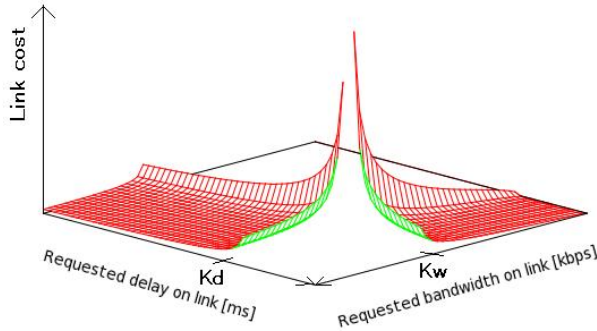


Fig. 2. Multi-variable cost function according to (30).

IV. THEORETICAL ANALYSIS OF THE NEW MULTI-VARIABLE COST FUNCTION

Conventional cost functions can be divided into two groups. The first group comprises of single-variable cost functions such as (1), (2), (3), (4), and (5). The other group comprises of multi-variable cost functions such as (6).

Fig.1(a) shows the value of a bandwidth-type single-variable cost function versus the requested bandwidth. Since $x_w < \kappa_w$, we only consider the left domain of the graph. We can see that, when the requested bandwidth approaches κ_w , the link cost increases rapidly to infinity. Fig.1(b) shows the value of a delay-type single-variable cost function versus the requested delay. Since $x_d > \kappa_d$, we only consider the right domain of the graph. We can see that, when the requested delay approaches κ_d , the link cost increases rapidly to infinity. Nevertheless, two single-variable functions failed to consider other important QoS parameters simultaneously. Fig.2 shows the value of the newly proposed multi-variable cost function. In this graph, only two QoS parameters (requested bandwidth and delay) are shown for demonstrative purposes. We only use the right part of the graph where $x_w < \kappa_w$ and $x_d > \kappa_d$. The main difference is that, the link cost will only increase rapidly when both requested bandwidth x_w and delay x_d exceed the maximum resources κ_w and κ_d . When only a single parameter exceeds its limited resource, a high cost will be assigned, however, it should not be high enough to block the entire link since we still have resources to assign for the other parameter. For example, when x_d exceeds κ_d , the cost increases to a high value, however, if we still have much bandwidth to assign, then the increased link cost will not

be sufficient to block the entire link. The link cost will only increase rapidly if both the requested bandwidth and delay simultaneously exceed their available resources (which are the total resources that the link can provide). Compared to the heuristic multi-variable cost function (6), our new multi-variable cost function has many advantages. Firstly, it has considered the requested QoS parameters from the application instead of only the residual resource from the underlay network. This is a major advantage since application's requirements are usually varied. Secondly, although the scaling factors K_i allows us to assign different weights for different parameters, it is not clear how to assign values for these factors in practice, therefore, there will be implementation problems. Thirdly, it is really unclear how we can add all bandwidth, buffer and delay together without any unit impairment. From the above analysis, we can conclude that our new multi-variable cost function owns better characteristics than conventional cost functions.

V. CONCLUSION AND FUTURE WORKS

In this research, a new multi-variable cost function has been proposed. The mathematical derivation process has also been described in details so that one can apply it to obtain other multi-variable cost functions according to their specific requirements. The newly found cost function has considered dynamic requirements of the application and the underlay network. For future works, intensive simulations will be performed to show the advantages of the newly proposed multi-variable cost function. A new ALM can be designed based on simulation and analysis results. The result can be further applied to improve the performance of any ALM algorithms who are using conventional cost functions to build their data delivery tree.

REFERENCES

- [1] S. E. Deering and D. R. Cheriton, "Multicast routing in datagram internetworks and extended LANs," *ACM Transactions on Computer Systems (TOCS)*, vol. 8, no. 2, pp. 85–110, 1990.
- [2] C. Diot, B. N. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," *IEEE Network*, vol. 14, no. 1, pp. 78–88, 2000.
- [3] M. Hosseini, D. T. Ahmed, S. Shirmohammadi, and N. D. Georganas, "A survey of application-layer multicast protocols," *IEEE Communications Surveys & Tutorials*, vol. 9, no. 3, pp. 58–74, 2007.
- [4] I. Matta and L. Guo, "On routing real-time multicast connections," in *IEEE International Symposium on Computers and Communications, 1999. Proceedings, 1999*, pp. 65–71.
- [5] D. H. Lorenz, A. Orda, and D. Raz, "Optimal partition of QoS requirements for many-to-many connections," in *IEEE INFOCOM*. Citeseer, 2003, vol. 3, pp. 1670–1679.
- [6] D. H. Lorenz and A. Orda, "Optimal partition of QoS requirements on unicast paths and multicast trees," *IEEE/ACM Transactions on Networking (TON)*, vol. 10, no. 1, pp. 102–114, 2002.
- [7] R. Widjono, "The design and evaluation of routing algorithms for real-time channels," *International Computer Science Institute, TR-94-024*, 1994.
- [8] A. Bueno, P. Vila, and R. Fabregat, "Multicast extension of unicast charging for qos services," in *Proceedings of 4th IEEE European Conference on Universal Multiservice Networks (ECUMN)*. Citeseer.
- [9] Le Tien Anh and Nguyen Hang, "Toward building an efficient application layer multicast tree," in *2010 International Conference on Research, Innovation and Vision for the Future*, November 2010.