VCAT Differential Delay Minimization for Delay Sensitive Multiservice Networks

Rohith Ramkumar, H.A.Chan, Senior *Member, SMIEEE*

Abstract— This paper focuses on eliciting a simulated cost effective algorithm to better facilitate the transport of delay sensitive traffic types across future Next-Generation Multiservice core networks. It examines the Multiservice paradigm and isolates the VCAT standard addressing the current issue of largely incessant differential delays faced whilst creating virtual paths through the network. These large differential delays amount to large and costly high speed buffers that are required at the endpoints on the network in order for the system to function acceptably. By observation of the current trends, as the volume of traffic on these networks increases with time, the increasing buffer sizes will eventually lead to unacceptable network maintenance costs as well as large physical property/estate required to house the buffers in the network nodes themselves. The paper thus explores the problem in detail and poses a solution based on the simulation findings to minimize the incurred differential delays. This finding thus provides an option by which the cost and housing space of the required buffers may be minimized, thereby increasing the maintainability of the networks themselves.

I. INTRODUCTION

Currently Next Generation SONET/SDH Multiservice core network architectures are capable of transporting a diversified set of communications standards encapsulating voice, video and data services. They have endured large changes in capability that have been facilitated through clever protocol usage. Their previous drawbacks of being rigid architectures capable of primarily transporting TDM voice based traffic have been, for the most part re-engineered by the introduction of Generic Framing Procedure (GFP), Link Capacity Adjustment Scheme (LCAS) and Virtual Concatenation (VCAT) optical protocol standards. These standards support features like increased link utilization, dynamic link upgrade, frame delineation etc. which are all needed in order to facilitate the transport of bursty data traffic over TDM-like systems.

In order to improve the efficiency and reduce the cost of maintaining these optimized Multiservice networks that we see in use today, it is imperative to address the issues that arise as a result of the increasing volumes of data that are passed through them. The issue at hand is that at present, there is more extensive usage of Multiservice networks which are being deployed to form the transport core for a wide variety of dynamic high bandwidth-on-demand application content. The result is that there is an increase in network cost and a decline in network performance and utilization which become harder to mitigate as the size of the network increases.

To address these issues, this paper puts forward an abstract design for Next-Generation Multiservice Networks and explores and evaluates an algorithm by which the transport of delay sensitive information will be enhanced, all of which culminate into a network that provides a maintainable and cost effective option that will suit the needs of this demanding middle ground environment.

II. NEXT GENERATION MULTISERVICE NETWORK ARCHITECTURE

A. *A Need for Resolve*

The process of evolution demands that there be a “fundamental need” that must be exist and continue to co-exist during the process to provide stimulus for the change. This was the case when legacy networks of past decades were subjected to huge changes in the dynamics of bandwidth demand. The shift to bursty data traffic was becoming eminent and the need was there to change the ineffective voice based SONET/SDH TDM architecture that represented the core transport network to one that could facilitate the transport of a varied set of traffic types with an assured QoS profile. This was the birth of the Next-Generation Multiservice networks.

B. *The Evolutionary Propensity*

At about the turn of the millennium, it became obvious that core networks would become a bandwidth intensive network zone that required smart provisioning, aggregation and switching architecture that would transport data to/from the increasingly variable access technologies being deployed at the edge of the network.

This process of evolution would be two-fold, as the legacy networks which existed, characterized by a plethora of Digital Cross Connects (DCS) and Add Drop Multiplexers (ADM) which were plagued by incessant complexity and localized bottlenecks were removed and replaced with a new “Multiservice-specific” node, the Multiservice Provisioning Platform (MSPP) which was capable of handling the bandwidth demands being experienced. In short, the MSPP’s were and still are complex centralized systems that are capable of transporting datacom and telecom services across dissimilar networks for varying traffic types between multi-protocol based endpoints. The systems were backward compatible with legacy equipment to facilitate a smooth transition towards a

Manuscript received May 1, 2006. Rohith Ramkumar is with the Electrical Engineering Department, University of Cape Town, Cape Town, South Africa, MSc student at the Communications Research Group, (e-mail: rohith.ramkumar@gmail.com).svs

Prof. H.A. Chan lectures for the Electrical Engineering Department and Supervises members of the Communications research Group, University of Cape Town, South Africa, SMIEEE.(e-mail: achan@ebe.uct.ac.za)

more manageable and configurable transport core. The MSPP with its ability to leverage the emerging Next-Generation Protocol standards: LCAS, GFP and VCAT enabled flexible and elastic transport of data and voice. The second stage of this process, as recommended by this paper is to enhance the working solution, mentioned above by extending its capabilities through the use of DWDM far into the future with the addition of 2 additional technology specific elements, the Multiservice Switching Platform (MSSP) and the Multiservice Transport Platform (MSTP).

C. *Elementary Triple Play Functionality*

The "Triple Play" Multiservice architecture is described in this section with attention given to where each of the multiservice elements mentioned above fits in. The overall architecture is to facilitate effective transport of voice, data and video and represent the future of core network technology well in to the future. Each emerging element has a specific function to perform as described below.

The MSSP is the transport cross-connect that provides efficient traffic grooming and switching to more efficiently pack traffic as it transits multiple switching centers. The term MSSP itself has been loosely associated as an MSPP of a grander implementation scale. It has generally been recognized that the term MSSP is used for MSPP's of switching capacities that *exceed 300 Gbps*. The MSSP is currently deployed "in between" and as on-ramps to core transport networks from the access rings.

The MSTP is the next generation transport element that combines the MSPP technology with DWDM capability. Previously MSPP technology was used to drive DWDM capability in the network but this sacrificed great amounts of switching fabric for DWDM integration. These enhanced DWDM capabilities of the MSTP means several benefits to the network, starting with the fact that it provides an alternative to using MSPP's with DWDM capability. At present DWDM capability is achieved through the use of a DCM module, an amplifier and a passive coupler placed in an MSPP whilst sacrificing expensive switch fabric.

The MSTP element is superior for large scale networks, and in the end leads to better space, cost and power efficiency when compared to an MSPP architecture which attempts to perform everything on a single platform. Long-haul networks tend to favor the integration of DWDM functionality because of the efficient use of the fibers and better optical bypass capabilities at intermediate nodes. Its is a more flexible element that uses DWDM to provide dynamic allocation of bandwidth in network zones that experience bandwidth demands that cannot be predicted a priori.

D. *Triple Play Implementation*

The "Triple Play" implementation is a relatively new concept that integrates the various NG network elements in a cohesive and competent manner in order to synergize the multiservice network design. Figure 1. below shows how the design emphasizes the balance between a high bandwidth capability and the ability to scale its bandwidth capability when it is in need of greater capacity.

From Figure 1, it becomes apparent that for the transport of delay sensitive traffic to be effective, the switching architecture needs to be efficient and remain efficient even under high load conditions. It is the VCAT standard that is responsible for inverse multiplexing schemes by which the switching of traffic is governed. The major issue that needs to be addressed now in this design is in eliciting more efficient ways by which to have dynamic connection management implemented into the system.

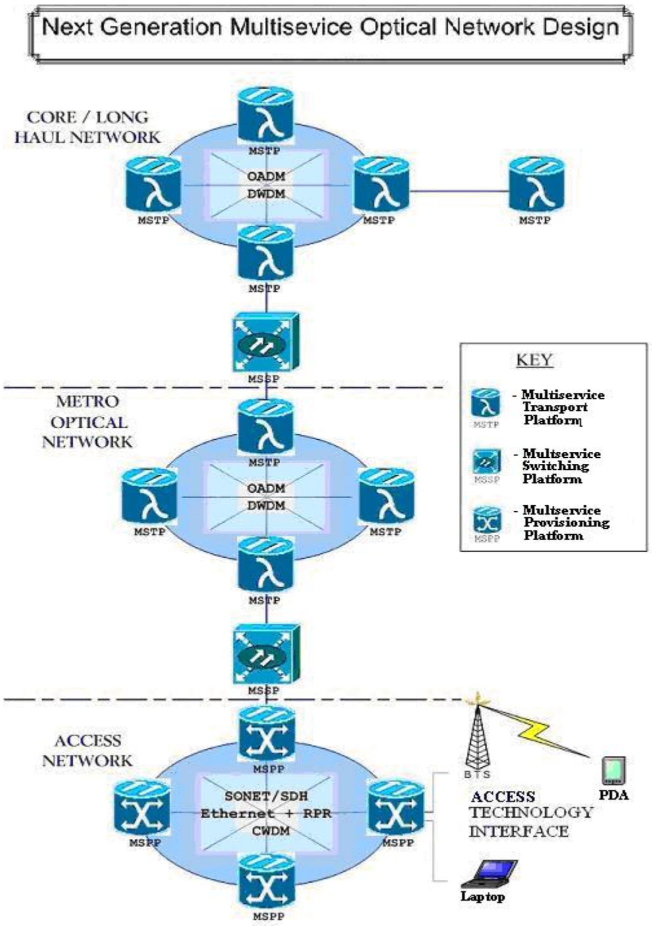

Figure 1 – The Triple Play Multiservice Architecture

Having mentioned this design, the issue this paper focuses on is in how the VCAT standard creates its virtually concatenated tributary paths and how it may be optimized in order to provide fast, reliable and cost-effective endpoint communications.

III. Physical And Logical Architectures

Before proceeding towards solving the task at hand it is important to understand the distinction between physical and logical network architectures that are concurrently in place performing specific functions in the network.

Every optical network is essentially a fixed point-to-point structure that may transport a given set of tributary carriers or frequency modulated signals that are given as inputs up to the maximum bandwidth capacity of the network. Physical networks are for the most part static and their endpoint connections remain fixed until they are changed, logical networks may have reconfigurations made in a short period of time. An example of a logical channel would be a virtual tributary channel in a given VCG (described further in the following section). Each tributary channel can be cross-connected to form varying patterns of logical connectivity. This concept is firmly rooted in the explanation of the following section on leveraging the flexibility of VCAT.

IV. UNDERSTANDING VIRTUAL CONCATENATION

A. VCAT Basics

Virtual concatenation (VCAT) is an inverse multiplexing protocol standard used on optical endpoints (MSPP, MSSP and MSTP), that facilitates dynamic logical link upgrade and on-demand bandwidth adjustment without the need for any hardware. In VCAT several optical signal carrier STS-n channels may be concatenated to provide an aggregated transport of a larger traffic load. The channels are commonly referred to as virtual tributaries (VT), and several VT's make a Virtually Concatenated Group (VCG). The multiple VT's that split the bandwidth requirement for transmission at the optical source node may be multiplexed at the source for information delivery to the destination node via different logical and physical paths. For example, 21 virtually concatenated STS-1s, denoted by STS-1-21v, can be used to provision a Gigabit data connection. Compared to the STS-48c bandwidth needed to carry this connection without VC, only STS-21bandwidth is needed with VC, resulting in a huge capacity savings (of approximately 60%).

As a result of this ability to inverse multiplex these VT's, network performance and efficiency can be improved. For example, link outages may be mitigated in high network traffic situations because several paths, that take less "busy" routes may be spawned to facilitate transport. This ability to enhance performance and efficiency is based on the assumption that the destination node may be able to buffer the incoming VT's such that concurrent and synchronous data delivery may be made possible for use of higher protocol layer applications such as VoIP.

Since each endpoint connection may have several VT's spawned to facilitate the connection there is simply no doubt that buffering each and every VT, for each and every endpoint connection in a high bandwidth environment requires huge amounts of high speed buffer space to allow the advantages of inverse multiplexing to be fully leveraged. This is why it is necessary to minimize the differential delay when recombining data streams from each virtual tributary.

B. The Origins of Differential Delay

The following section details how the VCAT protocol transports data and introduces differential delay from a theoretical and mathematical standpoint.

In the VCAT protocol, data are byte-interleaved over the multiple channels in the VCG, i.e. if there are X STS-n channels in a VCG, then the data are transmitted such that if the first byte is transmitted on the *r*th STS-n, then the next byte is transmitted on the (*r*+1)th STS-n, and so on until all X STS-n channels are transmitting data.

The key element here is at the receiver end, in that the sink node is responsible for reassembling the original byte stream. This responsibility includes compensating for the differential delay between the different network paths taken by the STS-n channels of a given VCG. The compensation of such delay is implemented by the use of a very high speed memory buffer. It is only when all frames of all the STS-n channels associated to the time-stamp have arrived at the receiver that the data packets can be extracted from them.

The last part of the above mentioned VCAT inverse multiplexing process, is the problematic section, where large external buffers are required to complete the transmission, a diagram of the problem is given below in Figure 2.

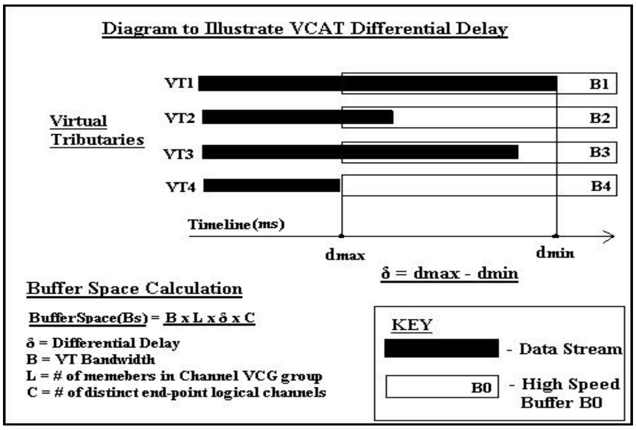

Figure 2 – Illustration of VCAT Differential Delay

The diagram above shows 4 incoming VT's on the left into 4 endpoint high speed buffers (B1-B4), the values d_{max} and d_{min} are illustrated and their difference δ is the differential delay. The buffer space (Bs) required in maintaining the communications between the endpoints is modeled in the equation given in Figure 2. It takes into consideration that there may be more than one channel that may be provisioned by way of the scalar C. It should also be clearly noted that the differential delay, δ is directly proportional to the buffer space (Bs).

In order to arrive at an acceptable technique to minimize the differential delay, we will need to analyze the problem further mathematically, the next section is dedicated to this.

C. Mathematical Representation of Differential Delay

Let $O = (V, L)$, represent a network O, with V distinct nodes and L separate links between distinct nodes. O is a directed network of constant parameters, with a fixed number of nodes V and links L. O has been modeled as an acyclic network, as seen by the network administrator of a given endpoint. Let a path $P_{u(i,j)}$ represent a distinct path $u \in N$ (positive integers), with i and $j \in V$. Therefore, given this description $P_{u(i,j)}$ can be seen as a set of interconnected nodes between the data source node i and the data sink node j. Eg. $P_{1(1,4)} = \{1,2,4\}$ and $P_{2(1,4)} = \{1,3,4\}$as shown in the network example given below. When the notation P_y is referred to it refers to a single distinct path between 2 pre-specified nodes. In the network below the nodes are numbered distinctly.

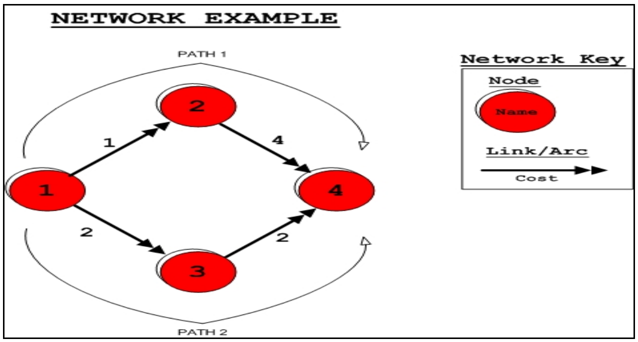

Figure 3 - Network O = (4,4). P1(1,4) = {1,2,4} and P2(1,4) = {1,3,4}

The depth of a node, D may be defined is the distance in number of links from the root node to the node of interest.

For example depth = 1, from node 1 refers to nodes 2 and 3 and depth = 2 refers to node 4.

In the network above let $L_{(i,j)}$ represent the arc/link cost between the nodes *i* and *j*. In the example above $L_{(2,4)} = 4$. When the notation L_y is used it refers to the arc/link cost of a distinct path between 2 nodes on a network.

Between links 1 and 4 let P_1*'s delay* $= 5$ $(L_{(1,2)} = 1 + L_{(2,4)} = 4)$ and P_2*'s delay* $= 4$ $(L_{(1,3)} = 2 + L_{(2,4)} = 2)$ as shown above. Then the description for differential delay may be given as below.

Definition of Differential Delay: Let $\zeta = \{P_1, P_2, P_3, \ldots P_n\}$ be the set of all distinct paths in a given network between two distinct nodes. $|\zeta|$ represents the total number of distinct paths between two distinct nodes. Let $d_{max}=(L_1 + L_2 + L_3 + \ldots L_n)_{Pmax}$ and $d_{min}=(L_1 + L_2 + L_3 + \ldots L_n)_{Pmin}$ be the cumulative delay of the highest and lowest delay paths in ζ respectively. Then, the differential delay is defined as follows:

$$\underline{\delta = d_{max} - d_{min}}$$

Now that we have a formal description, we may now create suitable path selection algorithms in order to attempt to minimize the differential delay, what follows it the logical process towards finding such a solution through detailed simulation.

V. VCAT SIMULATION EXPLAINED

A. The Simulation Criteria

In order to create a simulation test bed from which to evaluate a solution we need to look at why the system is inefficient and then find the algorithm that will be a suitable solution to the current problem. In eliciting this algorithm it is assumed that the delays of each link are known prior to evaluating exactly which set of paths is the optimal set towards minimizing the differential delay. Given that there exists several distinct physical paths between two interconnected nodes on a network, then there exists the possibility that VT's may be spawned across these links. Since the VCAT standard is not network aware, then the probability that a Virtual Tributary is spawned across a given link is equal for all links on that network. In this case it is also possible that two of the physical paths from the data source node to the data sink node which represent the longest and shortest delay paths may be used in the transfer of information between the endpoint nodes. The differential delay in this situation is the largest differential delay that may result from VT's spawned on a network, and is caused by the random path spawning process, which is how the VCAT standard currently operates. When such a situation occurs then the term MaxDD is given to that particular differential delay value, which may in fact be happening in real VCAT networks of today.

Several potentially viable algorithms were postulated, and eventually the k shortest paths algorithm showed the most promise. The problem itself is a two-sided constrained problem where one must try and minimize the mean delay as well as the differential delay across k distinct paths. The next section explains how the k shortest paths algorithm was implemented to evaluate the best possible paths to minimize δ.

VI. VCAT K SHORTEST PATHS SELECTION SIMULATION

The K shortest paths algorithm simulation was coded using C++, the differential delay problem was modeled based on mathematical graph theory. The algorithm works by exploring a graph, from a given source node outwards till it discovers the sink, it does this till all paths have been elicited and logged, then selecting the best K paths that minimize the differential delay. This implementation however, had a complexity of O(Kmlog(Kn)+ K_2m), of n nodes and m links, meaning that the algorithm is dependant on the actual size of the network that is being simulated. This complexity meant that the problem required a fair bit of generalization in order to produce valid results that could serve to provide relevant insight into the differential delay problem. The generalizations made are mentioned below.

1) The networks are fixed, with V and L constant with delays known before hand. There can be no dynamic resizing of networks, just as in real network conditions.

2) The networks are made acyclic and as symmetric in nature as possible. An example of a symmetric network is shown in the Figure 4 below. Ideal symmetric networks are networks in which node and link configurations are made identical from all subtending nodes of the root node.

3) The link delays used in the simulation are made random and between the range of 1-5 ms.

4) The following two simulation stages have been created assuming the fact that we are provisioning a Gigabit link between two endpoints with a STS-3c-7v connection involving 7 x 155 Mbps channels (K = 7) that need to be spawned.

Two specific simulations were made based on these generalizations. The first simulation (A) kept the number of links in the network constant whilst varying the number of nodes. The second simulation (B) kept the number of nodes constant whilst varying the number of links. Where possible the computational time and differential delays were gathered and compared to both the current VCAT random spawning technique as well as the MaxDD, possible in those network conditions.

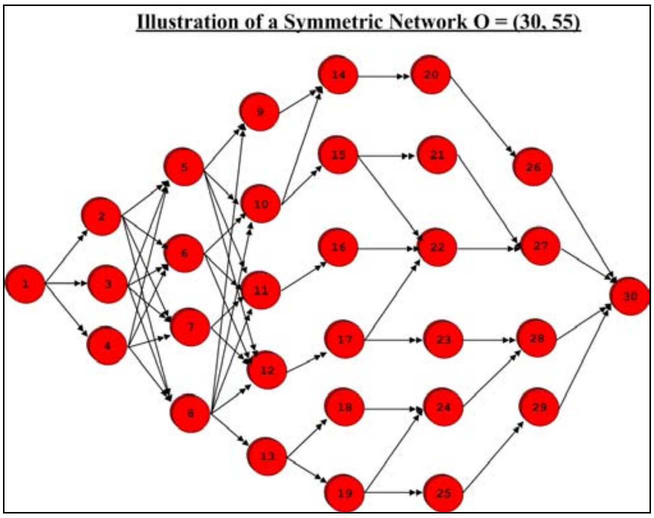

Figure 4 - Illustration of a Symmetric Network

A. *Simulation A Results*

Table 1 - Simulation A Results – Constant L = 55

Nodes [V]	# of Paths |ζ|	K Shortest Paths D.D.	Random Paths D.D.	Maximum D.D.
15	263	3	11	15
20	288	1	8	16
25	132	2	4	11
30	84	1	10	13
35	81	4	12	16
40	25	2	5	9
45	15	6	12	12

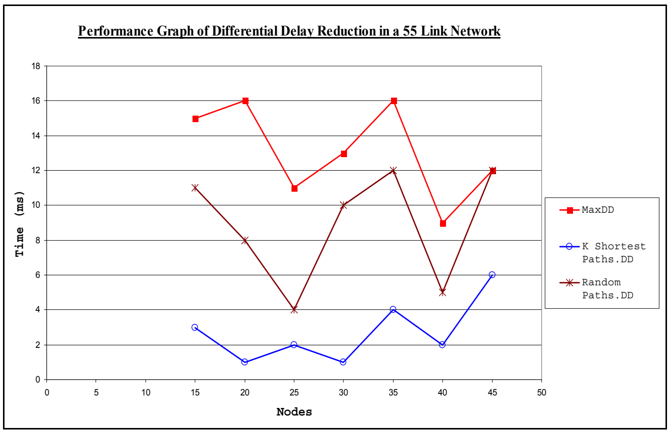

Figure 5 – Simulation A - Graph showing Differential Delay Results of a network of L = 55

From the results above, it clearly shows that the K Shortest Path algorithm facilitates the discovery of the 7 paths with the minimum differential delay. This is fairly intuitive, but what is impressive is the magnitude to which it scales down the differential delay. From the graph, on average the K Shortest Path DD is about 21% of the Max DD and is again on average only 30% the Random Differential Delay value. This result, showing a three-fold optimization in differential delay is a strong driver in suggesting its implementation in VCAT enabled systems. Since the differential delay is proportional to the actual buffer size, then it can be said that by downsizing the differential delay to 30% of it original value (Random Paths) we are effectively downsizing the buffer requisite to 30% of it original value as well.

It could also be mentioned that as the number of nodes increased the Random Paths DD tended to coincide with the values of Max DD, which is a sign of the inefficiency of the Random Paths algorithm which is what is existent with the VCAT technology at present.

B. *Simulation B Results*

Table 2 - Simulation B Results - Constant V = 30.

Links [L]	# of Paths [|ζ|]	K Shortest Paths D.D.	Random Paths D.D.	Maximum D.D.
50	56	3	8	21
100	576	2	4	19
125	1746	1	7	19
150	5120	2	7	13

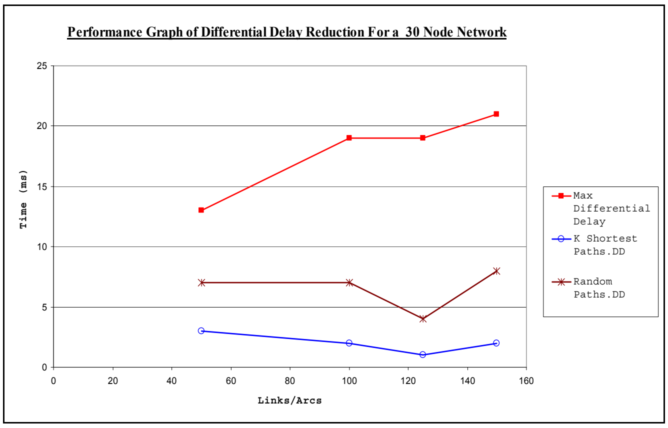

Figure 6 - Simulation B - Graph showing Differential Delay Results of a network of V = 30

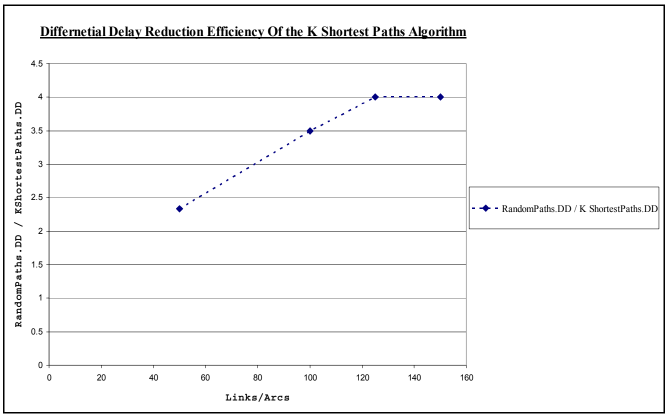

Figure 7 - Graph illustrating Comparative DD Minimization Efficiency of K Shortest Paths algorithm to Random Path selection

Again in Simulation B, it can be seen that the K Shortest Path provides the best differential delay and on average reduces to about 11% of the Max DD (Figure 6), a metric that represents increased network reliability. It should be noted that the minimization variance between the two simulations A and B suggests that the K Shortest Path algorithm tends to optimize the differential delay more efficiently in larger networks.

The K Shortest Paths algorithm again finds paths that minimize the differential delay on average to about 30% of what the Random paths selection achieves, which in turn translates to roughly a 3-fold decrease in buffer space. Figure 7 compares the differential values of the K Shortest Path and Random Selection algorithm and it goes to suggest an increasing optimization trend of the K Shortest Path algorithm as the number of links and thus paths in the network increase. The graph's Y-axis represents the value of the Random Path differential delay divided by the K Shortest Path differential delay in order to quantify how many times better the K shortest path algorithm is than the Random Path Selection algorithm. This graph also elicits a vital statistic, that on average, the head to head ratio between the current VCAT implementation (using Random Path selection), and the proposed technique (based on the K Shortest Path) algorithm will in effectively reduce the buffer space to anywhere between 25 to 45 percent of what is currently being used. This represents a truly outstanding benefit to VCAT enabled systems.

Table 3 - Simulation B Algorithm Process Runtime - Constant V = 30.

Links [L]	**Paths** [$|\zeta|$]	**K Shortest Paths Time (ms)**	**Random Paths Time(ms)**
50	56	7	7
100	576	161	105
125	1746	483	329
150	5120	1686	770

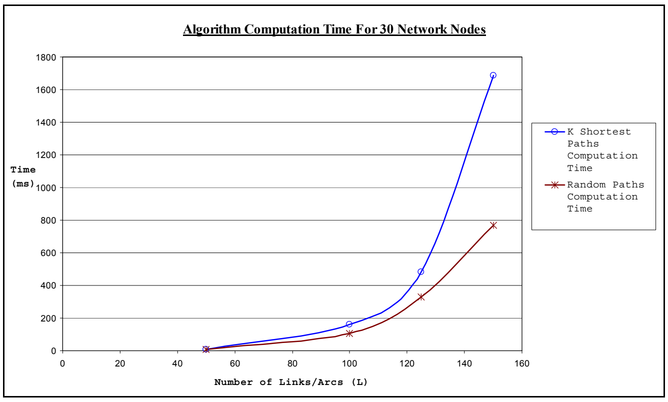

Figure 8 - Simulation B - Graph of Process Runtime of KSP and Random Path Algorithms - Constant V = 30.

The above graph suggests that as the number of links and associated paths increases the K Shortest Paths algorithm takes an exponentially longer time to compute as compared to the Random Paths Selection algorithm. This implies that in real networks that contain a larger number of paths, the algorithm will require powerful network processors to function in real time or will need to be used whilst provisioning transport services between endpoints prior to the start of transmission between VCAT enabled nodes.

VII. CONCLUSION

The findings from the simulations go towards suggesting that a huge 3-fold decrease (i.e. down to 30% of the original value of δ), is possible from the currently utilized random selection technique. This minimization is facilitated through the use of the K Shortest Paths algorithm which selects paths that minimize the differential delay. This 3-fold minimization in δ, translates to a proportional minimization in high speed buffer space required, cutting the cost of the buffer component down to 30% of its original value.

The algorithm evaluated in this research seems to work better in larger real life environments with its only drawback being its processing complexity and time.

There is yet room for improvement as searching/sorting techniques could be optimized to arrive at the best paths sooner meaning a more pragmatic real time solution.

Finally, it seems certain that highly efficient VCAT path selection algorithms will become a mainstay of true Multiservice networks of the future. The results from this paper further emphasize the value and importance of Multiservice networks, in particular of the Triple Play implementation which seems highly likely to become the way forward for transport core networks.

REFERENCES

[1] Vivek Alwayn - “Optical Network Design and Implementation”- Cisco Press, 2004

[2] Satyajeet S. Ahuja, Turgay Korkmaz, Marwan Krunz - “Minimizing the Differential Delay for Virtually Concatenated Ethernet Over SONET Systems” - University of Arizona.

[3] Mark Allen Weiss - Data Structures and Problem Solving in C++ - 2nd Edition - Addison Wesley Longman, 2000

[4] Intel – “Multiservice Provisioning Platform” – Available [Online] - http://www.intel.com/design/network/solutions/manual/Chapter19.pdf

[5] Ron Fang – “SONET, GE and CWDM in the evolution of First Mile Business Access” – Available [Online] - http://www.cse.ohio-state.edu/~jain/opticomm/ftp/2_1_fang.pdf

[6] Cisco Press – “Introducing DWDM” – Available [Online] - www.cisco.com/univercd/cc/td/doc/product/mels/cm1500/dwdm/dwdm_fns.pdf

[7] IEC Alcatel - “Optical Networks” - Available [Online] - http://www.coe.montana.edu/ee/rwolff/EE548/sring05%20papers/OPtical%20amplifiers/opt_net-tutorial.pdf
R. W. Lucky, “Automatic equalization for digital communication,” *Bell Syst. Tech. J.*, vol. 44, no. 4, pp. 547–588, Apr. 1965.

[8] Rajiv Ramaswami – “Future of Optical Networking” – Cisco Press http://networks.cs.ucdavis.edu/~mukherje/info03/infocom03_rajiv.pdf

[9] Turin Networks - “Redefining SDH” – Available [Online]http://www.turinnetworks.com/Redefining_SDH_White_Paper.pdf

[10] Wayne D. Grover “Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET, and ATM Networking”, Prentice Hall, 08/2003.

[11] Nortel – “Key Advantages of Next Generation SONET” – Available [Online] - http://www.nortel.com/products/01/optera/metro/msp/collateral/nn-100500-0502.pdf

[12] Anurag Srivastava, Swarup Acharya, Mansoor Alicherry, Bhawna Gupta, Pankaj Risbood - “Differential Delay Aware Routing for Ethernet over SONET/SDH” - Bell Laboratories, Lucent Technologies

[13] Fujitsu – “SONET and DWDM: Competing Yet Complementary Technologies For The Metro Network”

[14] Alicherry, M.; Phadke, C.; Poosala, V. “Delay distributed VCAT for efficient data-optical transport”, Optical Fiber Communication Conference, 2005. Technical Digest. OFC/NFOEC Volume 4, Date: 6-11 March 2005, Pages: 3 pp. Vol. 4

Rohith Ramkumar- Born in India, DoB: 12/07/1983 graduated in December 2005 with a BSc(Eng) Electrical and Computer (Telecommunications Centric) at the University of Cape Town, RSA. He is currently at the University of Cape Town pursuing his MSc(Eng) Electrical Engineering. Research Areas include: Core Networks, Survivable Networks and Multiservice Networks.