Dynamic Path Scheduling through Extensions to Generalized Multiprotocol Label Switching (GMPLS)

An SBIR Project with United States Department of Energy under Grant No. DE-FG02-06ER84515

Principal Investigator: Dr. Abdella Battou

Final Phase II Report

May 22, 2009

Executive Summary

The major accomplishments of the project are the successful software implementation of the Phase I scheduling algorithms for GMPLS Label Switched Paths (LSPs) and the extension of the IETF Path Computation Element (PCE) Protocol to support scheduling extensions. In performing this work, we have demonstrated the theoretical work of Phase I, analyzed key issues, and made relevant extensions.

Regarding the software implementation, we developed a proof of concept prototype as part of our Algorithm Evaluation System (AES). This implementation uses the Linux operating system to provide software portability and will be the foundation for our commercial software. To demonstrate proof of concept, we have implemented LSP scheduling algorithms to support two of the key GMPLS switching technologies (Lambda and Packet) and support both Fixed Path (FP) and Switched Path (SP) routing. We chose Lambda and Packet because we felt it was essential to include both circuit and packet switching technologies as well as to address all-optical switching in the study. As conceptualized in Phase I, the FP algorithms use a traditional approach where the LSP uses the same physical path for the entire service duration while the innovative SP algorithms allow the physical path to vary during the service duration.

As part of this study, we have used the AES to conduct a performance analysis using metro size networks (up to 32 nodes) that showed that these algorithms are suitable for commercial implementation. Our results showed that the CPU time required to compute an LSP schedule was small compared to expected inter-arrival time between LSP requests. Also, when the network size increased from 7 to 15 to 32 nodes with 10, 26, and 56 TE links, the CPU processing time showed excellent scaling properties.

When Fixed Path and Switched Path routing were compared, SP provided only modestly better performance with respect to LSP completion rate, service duration, path length, and start time deviation. In addition, the SP routing required somewhat more CPU time than the FP routing. However, when the allowable range for the start time is increased, the CPU time for SP increased less rapidly than FP such that the CPU processing times became comparable. Therefore, SP routing may scale better than FP routing.

The Path Computation Element Working Group is a complementary working group to the GMPLS working group (CCAMP) in the IETF and is developing standards to discover, manage, and access elements that compute routes for GMPLS for LSPs. The algorithms used to compute the routes are not subject to standardization. Since the PCE supports only standard GMPLS LSPs, it does not support scheduled LSPs. Therefore, it is a natural extension of our Phase I work to introduce this enhancement into the PCE. In addition, the PCE adds another dimension to our product line because the PCE by itself may be a product with a standard interface.

With the results of Phase II, we are now positioned to begin commercialization, Phase III. We will continue our ongoing fund raising and customer interaction efforts to establish a basis of support for our efforts. After achieving this support, we will complete a networked PCE prototype, i.e., interconnected to a selected network element, and then enter the product development phase. Also with the support of our customers, we will standardize the scheduling enhancement to GMPLS and PCE protocols.

Table of Contents

List of Figures

List of Tables

1 INTRODUCTION

1.1 Overview

The purpose of this document is to present the results of the work performed under the Phase II SBIR project for the U.S. Department of Energy. This scope of this work has encompassed the extension of the Phase I Generalized Multi-Protocol Label Switching (GMPLS) work to introduce the Path Computation Element (PCE) with scheduling capabilities, a prototype implementation of the scheduling algorithms, the evaluation of these algorithms, and the formulation of plans for commercialization (Phase III).

The results of this work indicate that the implementation of the distributed scheduling algorithms is technically feasible. In particular, the algorithm implementation of the algorithmic techniques was straightforward, and the resulting algorithm execution during case studies was scalable with respect to network size.

1.2 Organization of the Report

The remaining sections of this document present:

- System Description - description of potential operational environments where the distributed scheduling environment may be deployed,
- Algorithms – description of the computational algorithms using both fixed and switched path techniques,
- Protocols – extensions to standard GMPLS and PCE protocols to support scheduled services,
- Performance Analysis – set of case studies evaluating the scheduling algorithms for all-optical and packet switching technologies,
- Phase III plans – follow-up activities envisioned for commercialization of the distributed scheduling technology,
- Bibliography – list of informative and normative references.

2 SYSTEM DESCRIPTION

This section describes how the distributed scheduling technology may be deployed in representative system environments. Note there is no one recommended (or preferred) system concept for deployment of the distributed scheduling technology. Rather there are a set of options available such that one may be selected for a particular operational environment. This description encompasses the following topics:

- Operations Concept describes the top-level interaction among key system elements using distributed scheduling,
- System Architecture describes the major system elements and their interconnection for representative deployments,
- System Services describes the distributed scheduling services provided to users,

- Functional Allocation describes the allocation of functions to system elements in order to provide distributed scheduling services.

In addition, this section describes the Algorithm Evaluation System (AES) used in this project to evaluate the performance of the distributed scheduling algorithms.

2.1 Operations Concept

The innovative feature of the Distributed Scheduling technology developed in this project is that all network elements are *time-aware of future resource allocation*. This feature enables a faster, more robust management of scheduled LSPs than using a centralized approach because LSPs may be set up, released, and recovered without the intervention of the Network Management System (NMS).

In this concept, the Path Computational Element (PCE) generates the schedule in response to a request from Path Computation Client (PCC) resident in an external element (either a Network Management System or Network Element). In addition to the normal GMPLS service parameters, this request includes a desired start time, T_S, and duration, T_D. As shown in Figure 2-1, the PCE then forwards the schedule consisting of the LSP actual (assigned) service start time, ts, and duration, td, to the ingress Network Element (NE) via the PCC. Then the Network Elements reserve the resources necessary for the LSP. At time ts, the ingress network element initiates the activation of the LSP without any intervention of the NMS. Then at time ts+td, the ingress network element initiates release of the LSP without intervention of the NMS.

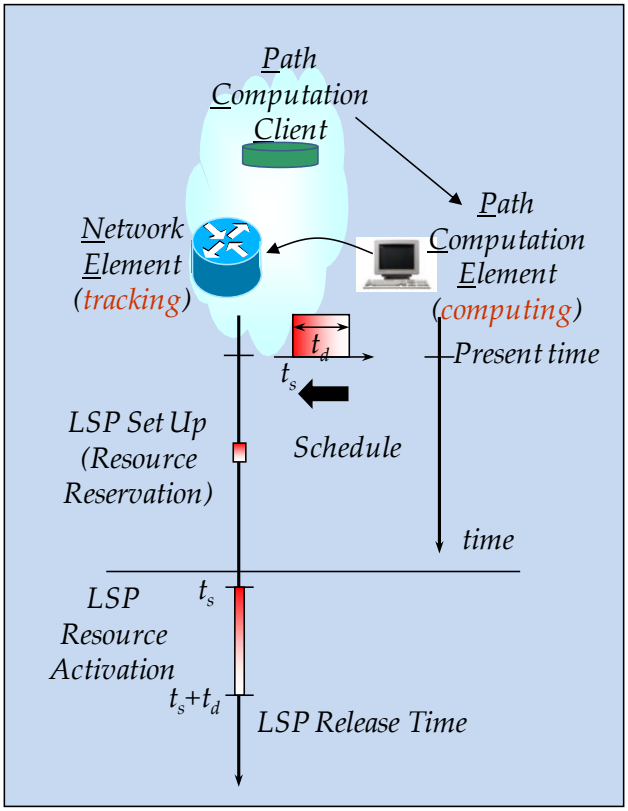

Figure 2-1: Distributed Scheduling Concept

With this approach, there are no messages between the control plane and the management plane at the time of activation so delays will be reduced and failure points eliminated during activation. Therefore, the activation may be done faster and more robustly.

Having the schedules stored in the control plane provides major advantages for recovery. When a network element is in a recovery mode, it will be able to retrieve the resource schedule from its neighbor using advanced GMPLS protocols. This involves Phase III work.

2.2 System Architecture

The distributed scheduling technology has been developed to use the framework provided by the standard IETF GMPLS protocol suite using RSVP-TE signaling [15] and OSPF-TE routing [8] and the related Path Computation Architecture [22]. While operators will have broad flexibility in using this technology in their environment, there are two primary deployment options depending upon the residency of the Path Computation Client (PCC).

The first deployment option has the PCC resident in the Network Management System (NMS). As depicted in Figure 2-2, this architecture consists of an Enhanced GMPLS Network, Path Computation Element (PCE) with a Traffic Engineering database (TED), and Network Management System (NMS). In this concept, the NMS requests the path-schedule from the PCE and provides the result (service start time, duration) to the ingress Network Element (NE) in its request to set up the path. Then the NE will set up the path reserving resources immediately (indicated by the dashed lines in the figure) and later activate the path (set cross-connects) at the assigned LSP service start time (indicated by the solid lines in the figure). The ingress NE provides the NMS of event notifications as resources are reserved and activated. Also, the PCE receives OSPF advertisements so that it may track loading on the TE links.

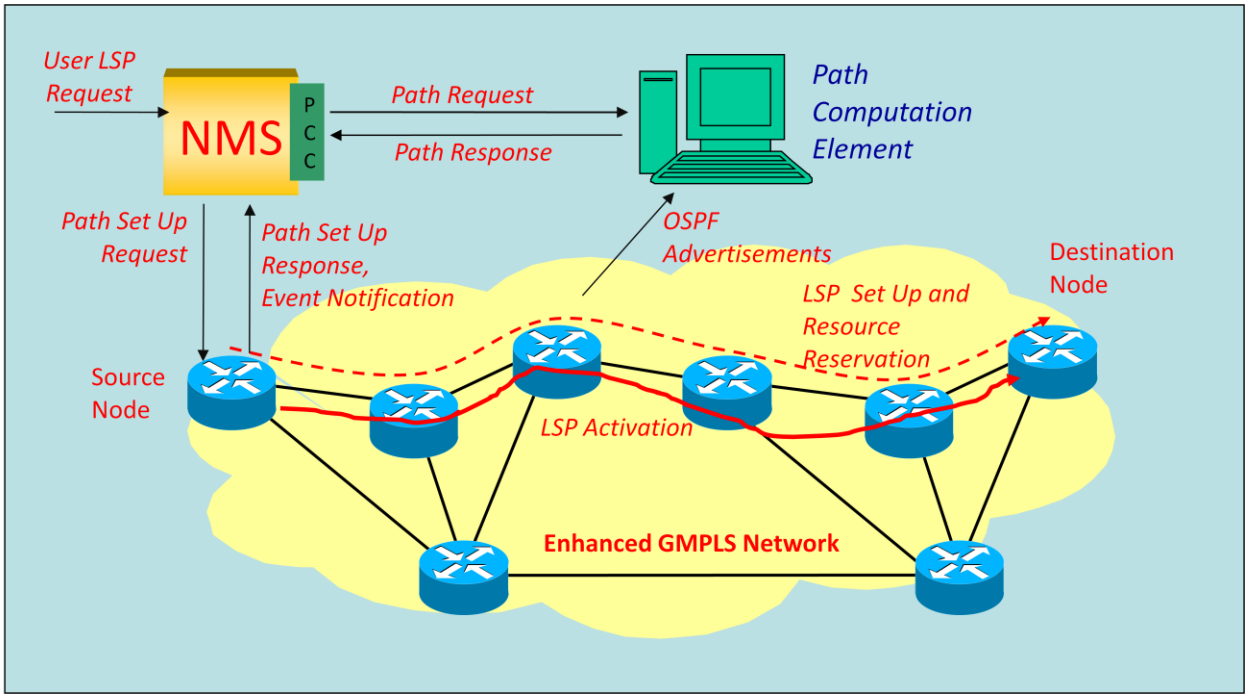

Figure 2-2: System Architecture – PCC Resident in NMS

In the second deployment option, the PCC is resident in each Network Element (NE). As shown in Figure 2-2, the NMS submits a request for scheduled path directly to the ingress Network Element and the request includes the desired service start time and duration. Then the NE will request the path-schedule from the PCE. Upon receiving the response from the PCE, the NE will set up the path reserving resources (dashed line in the figure). At the scheduled activation time, the NE will activate the path (solid line). These deployment options are equivalent from a user perspective and will be supported in future products.

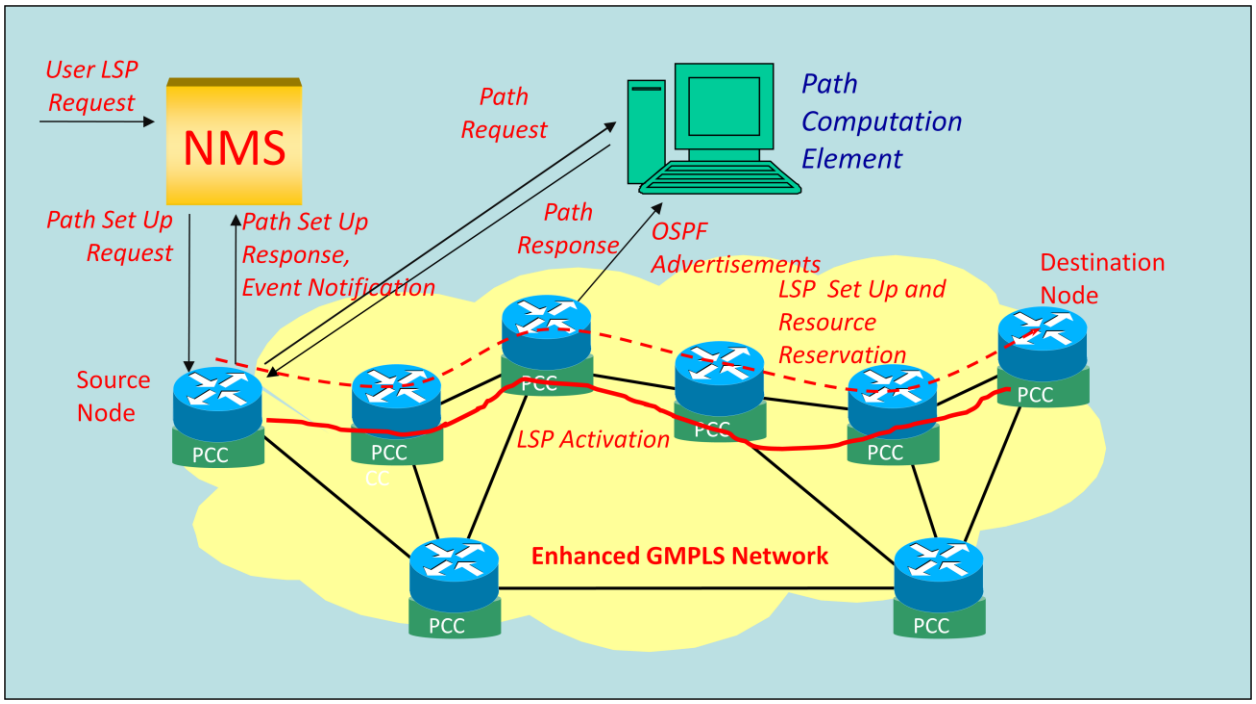

Figure 2-3: System Architecture – PCC Resident in NE

While not depicted in the figure a Data Communications Network (DCN) is also required for communication between the Network Elements, PCE, and the NMS. Either an internal DCN using in-fiber control channel on the DWDM fiber between the NEs (with external links providing connectivity to the PCE and NMS) or an external DCN may be used. The selection of the DCN has no impact on this research.

This work is applicable to all GMPLS switching technologies. The work in this project will focus on all-optical and packet switching. The extension to support other GMPLS switching technologies, e.g., opaque optical network, Ethernet, and TDM, is straightforward.

The PCE may be either stateful or stateless depending upon whether the PCE tracks the state of individual LSPs. The ongoing IETF standardization work is based on a stateless PCE because it is more scalable without having to track the LSP state. If the PCE is "stateless," this reduces the PCE to a *scheduling engine* in charge of *computing* schedules only, with *tracking* offloaded to the associated network elements. However, when operated in a "stateful" mode, the PCE also tracks individual LSP assignments to minimize conflicts in LSP set up.

For this project, the PCE is implemented as "stateful" in particular to facilitate enforcement of the wavelength continuity constraint in all-optical networks based on the LN 2000 all-optical switch. In subsequent development efforts, the PCE will be implemented as stateless because this mode is better suited to supporting very large number of LSPs typical of Ethernet networks.

2.3 Enhanced GMPLS Network Elements

The Enhanced GMPLS Network supports the standard GMPLS signaling protocol, RSVP-TE [14-21], and routing protocol, OSPF [7-13], with enhancements to provide scheduled services as defined in Phase I [1] and refined in this project. These enhancements allow the user to specify a desired start time and duration for the LSP service, parameters that are not supported by the standard GMPLS protocols.

The approach taken in enhancing the GMPLS protocols is a generalization of the existing GMPLS protocols by introducing schedule specific objects and operations. If a node does not support these enhancements, it will still be interoperable with the enhanced GMPLS nodes. Refer to Section 4 for details on interoperability with existing nodes.

When the Network Elements are implemented using an all-optical switching fabric in the Phase III target platform, the applicable GMPLS switching technology is lambda switching. One platform that implements this type of switching is the LN2000. However, for a hierarchical switching architecture like the LN2000, this architecture may support only scheduled services for wavelength switched LSPs but not band switched LSPs. The band switched LSPs may be configured, but in the future these LSPs may also be scheduled.

The NEs utilize transponder based interfaces supporting SONET/SDH 2.5G and 10G data encoding. Both fixed wavelength and tunable wavelength transponders are supported.

2.3.1 NMS

When the PCC is resident in the NMS, the NMS obtains LSP path and wavelength from the PCE and provides the path-wavelength in the request to the ingress NE to instantiate the LSP. For access to the PCE, the NMS implements the IETF Path Communication Element Protocol (PCEP) with scheduling enhancements. The standard PCEP is specified in [23]. Alternatively, the NMS may request the Network Element to schedule and set up the path. In this case, the PCC resides in the Network Element.

In addition, the NMS performs the overall management of both the Enhanced GMPLS Network and the PCE – Fault, Configuration, Accounting, Performance, Security, Management. When operating as a management system, it uses SNMP to communicate with the Network Elements and the PCE.

When the "stateful" mode is being used, the NMS also provides more detailed Traffic Engineering information using proprietary enhancements to the PCEP. This information includes detailed LSP information on LSP state.

2.3.2 PCE

The PCE is based on the IETF defined architectural framework [21] and uses the PCE Communication Protocol [23] to communicate with the PCC wherever it resides. While not being implemented in this project, other PCE standards that address discovery [24], policy [25],

operation with multiple switching layers [26], and SNMP management [27] will be implemented in Phase III.

As discussed above, the PCE implementation may be either "stateless" or "stateful." In the "stateless" mode, the PCE will only compute the schedules, but only the NEs are responsible for tracking their individual schedules. This may result in conflicts when back-to-back circuits are setup with very little time between circuit set ups. Also, the "stateful" mode operation may be needed to support advanced features such as LSP re-optimization and enforcement of the wavelength continuity constraint.

To address these issues, the initial PCE implementation will be "stateful" where it also tracks the state of all LSPs in the network. For example, PCE knows the LSPs that are currently being established. With this information, the PCE will not try to set up LSPs using the resources already assigned to LSPs but whose usage has not yet been advertised by OSPF. Also, the PCE will be able to track wavelength availability needed for wavelength continuity enforcement.

In response to LSP path requests from the PCC in the NMS, the PCE generates a path and schedule (starting time, duration) for the requested LSP. To generate the path, it will implement a suite of algorithms as defined in Section 3. The specific algorithm that is used in Path Computation is determined by user configuration.

In order to determine the network status for use in path computation, the PCE listens passively to OSPF Link State Advertisements. In addition for the "stateful" mode, the PCE obtains more detailed LSP connection and configuration data from the NMS for use in this computation.

To facilitate efficient path computation, the PCE maintains a Traffic Engineering Database (TED). This database consists of a structured representation of the network configuration, topology, and LSP information. It includes both the information obtained from the OSPF LSAs as well as from the NMS.

The PCE is implemented as a standalone, centralized Linux PC or workstation supporting one OSPF routing domain. To provide fail-safe operation, the centralized PCE would be implemented with primary and backup servers in the commercialization phase (Phase III). Since failure conditions are not being addressed in this research, only the primary server is implemented in this project. However, the same techniques that have been used to implement a redundant nodal processor in our commercial switching product can be applied to the PCE. It will support real-time operation with a switchover in less than 15 seconds after detection of failure.

2.3.3 Algorithm Evaluation System

Since the thrust of this effort is to implement and analyze the computational algorithms, only a standalone PC or Workstation running Linux will be used to provide a computational platform. As depicted in Figure 2-4, the platform will support the PCE with a TED as well as a Network Script to simulate the submission of user requests and network activity.

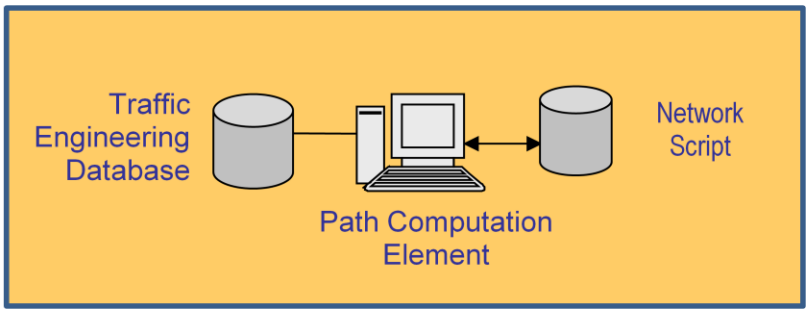

Figure 2-4: Algorithm Evaluation System

Computational algorithms have been developed and analyzed for both lambda switching and packet switching technologies. Therefore, the network script models both switching technologies.

2.4 Services

This section describes the LSP services, presents an example scenario, and describes the differences between switched path and fixed path service.

2.4.1 Overview

This primary service supported in this work is a scheduled service where the user provides a requested start time, T_S, and duration, T_D, in addition to the normal LSP parameters such as the LSP endpoint addresses, service level, and QoS parameters. It is assumed that all scheduled services have the same priority and are not pre-emptable. Given these parameters, the PCE generates a path that optimizes an objective function based on the path length, deviation from the desired starting time, and deviation from the desired duration.

In addition, on-demand services are also supported. In contrast to scheduled services, these services are set up immediately and have an undetermined duration, i.e., they are terminated in response to a release request. Also, on-demand services support GMPLS priorities, but they never pre-empt scheduled services.

In this project, the service level is assumed to be unprotected. However, the enhancements to support other service levels such as 1+1 protection and auto-restoration will be added in Phase III as market needs require these enhancements.

The QoS parameters are dependent on the switching technology. Both lambda and packet switching technologies will be evaluated in detail.

2.4.2 Scheduled LSP Services

This section presents a scenario illustrating the set up of a scheduled LSP originated by the NMS. Although it is possible to set up scheduled LSPs where a peer device such a router initiates a Lambda LSP request, it is more typical to initiate the set up through the NMS in response to a user request. For this scenario, it is assumed that the PCC is resident in the NMS.

Figure 2-5 depicts the representative LSP set up scenario for the "stateful" mode where the NMS initiates the set up and via the PCC requests a path-schedule from the PCE. Upon receipt of a response from the PCE, the NMS requests the ingress NE to set up the path. The NE will set up the path reserving resources immediately and later activate the path on its own.

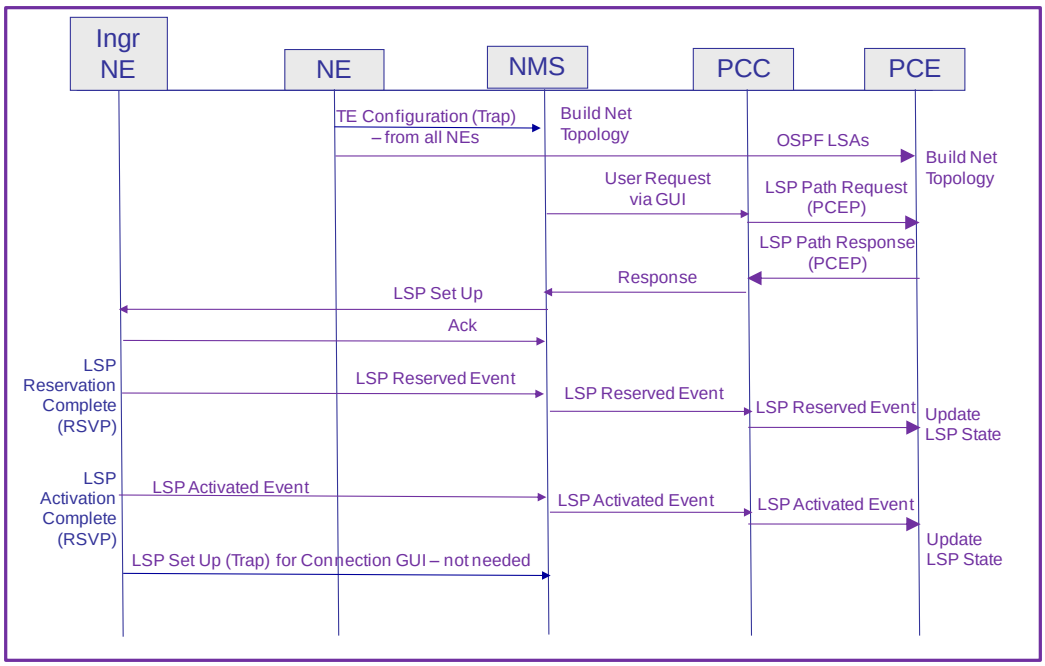

Figure 2-5: Scheduled LSP Set Up Scenario for Stateful Operation

As the LSP is set up and activated, NE informs the NMS. Then the NMS provides status updates to the PCE enabling the PCE to maintain the state of the LSP.

As shown in the scenario above for "stateful" operation, the PCE obtains:

- OSPF TE data derived from the Link State Advertisements (LSAs) received from the NEs including TE endpoint IP addresses, interface IDs, TE metrics, SLRGs, administrative colors,
- detailed TE data from the NMS providing the current status of each LSP.

When operating in the "stateless" mode, the PCE may use the available bandwidth advertised in the LSAs in lieu of the individual LSP state. This approach may be satisfactory for networks using switching technologies such as packet or Ethernet.

2.4.2.1 LSP Request and Path Computation

Using the PCE Communications Protocol (PCEP) [20], the PCC in the NMS will request an LSP path from the PCE specifying the desired start time, desired duration, LSP endpoints and relevant QoS parameters. Then the PCE will compute the path-schedule using one of the computational algorithms specified in Section 3. It will provide the computed start time, computed duration, and path to the NMS.

In this project, the PCE provides only one path-schedule. However, the algorithms may be enhanced to provide a schedule set (list of start times, duration, and paths) allowing the NEs to negotiate the actual schedule.

2.4.2.2 LSP Set Up

After the PCE returns the path to the NMS/PCC, the NMS will set up the LSP by sending an SNMP command to the source NE including parameters to fully define an Explicit Route Object

(ERO) including the wavelength on the DWDM links. Loose EROs will be handled in Phase III, e.g., support for inter-domain requests.

After the source NE receives the SNMP from the NMS, it will initiate the RSVP-TE signaling to establish the LSP. After the LSP set up is complete, the NE notifies the NMS using an SNMP trap. The NMS will then notify the PCE as part of the detailed TE data (stateful mode). Similarly, when the LSP is activated, the NE will notify the NMS who will in turn notify the PCE.

2.4.3 Switched Path and Fixed Path Service

Scheduled services may be implemented as either fixed path services or switched path services. When the service is implemented as a fixed path service, the path supporting the service does not change throughout the duration of the service. This option minimizes software complexity.

However, for a specific LSP request, a fixed path may not exist in the network given the current network loading. Also, the PCE may generate shorter paths if the network is allowed to re-route paths during the service duration. Therefore to reduce blocking and increase network utilization, the system also supports switched paths where the network path is re-routed while the service is active. For switched path LSPs, all of the paths are generated at the time of circuit request.

Figure 2-6 illustrates the LSPs that were established for the interval (t1, t2) in the order LSP1, LSP2, LSP3, and LSP4. Assume LSP1 was forced to take the long path due to the setting of the link metrics. Also, assume the bottom path is shorter for LSP2 given the link weights. In this case, LSP2, LSP3 and LSP4 are longer than necessary.

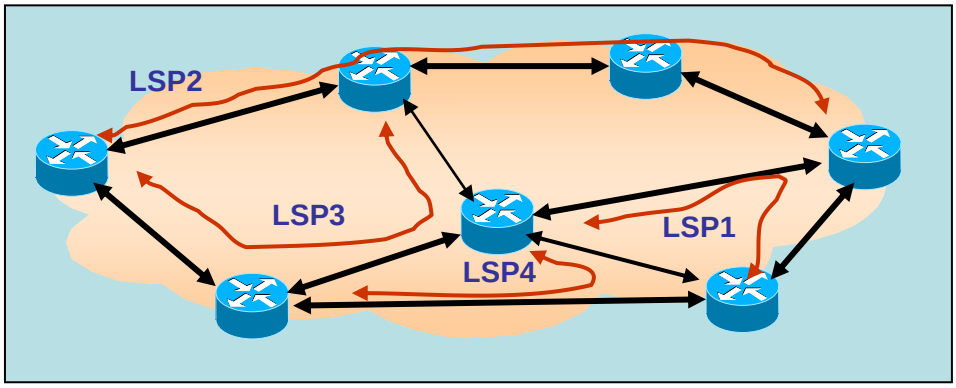

Figure 2-6: LSP Initial Routing during interval (t1-t2)

For a re-routing scenario, assume that LSP1 terminates at t2 while the other LSPs persist through t3. Then at LSP set up time, the PCE would be able to obtain shorter paths for LSP2, LSP3 and LSP4 for the interval (t2, t3) as shown in Figure 2-7 because the PCE knows LSP1 will be released at time t2.

However, the switchover at time t2 will result in a service disruption for optical circuits and service degradation for Ethernet LSPs because of the resource dependency of the LSPs. Specifically, the re-routing introduces a circular dependence:

- LSP2 requires resources being used by LSP4,
- LSP3 requires resources being used by LSP2,

- LSP4 requires a resource being used by LSP3.

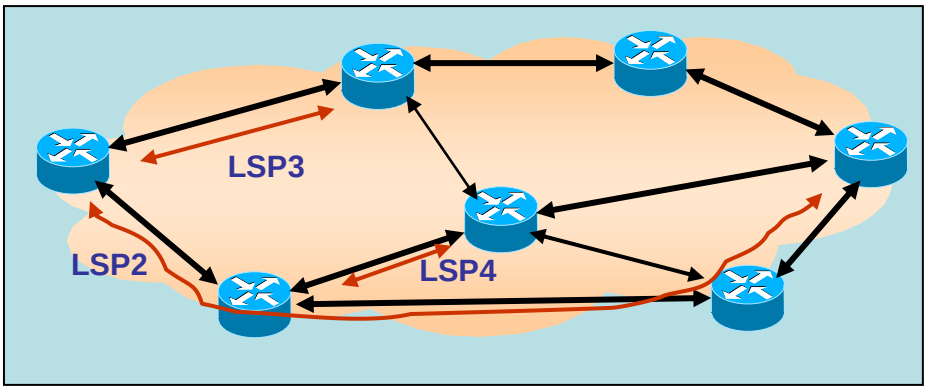

Figure 2-7: LSP Routing during interval (t2-t3)

With this circular resource dependence, it is not possible to select a sequence of re-routing that will not impact the QoS of at least one circuit – even if make before break signaling is used.

2.4.4 On Demand LSPs

As discussed above, on demand LSPs are also supported in the Target System. The major complication in supporting scheduled and on demand is the allocation of bandwidth. Two options are provided:

- Integrated – scheduled and on demand LSPs share the same bandwidth,
- Partitioned – scheduled and on demand services have their own dedicated bandwidth allocations.

These options affect the protocols and path computation. If the network is using the partitioned option, OSPF needs a mechanism to indicate the TE links corresponding to the individual allocations and to properly update the available bandwidth after LSPs are set up or released.

When bandwidth is shared, pre-emption of scheduled LSPs by on demand LSPs (and vice versa) is not allowed.

2.5 Software Functionality

This section defines the software functionality for each system element. For the PCE, it presents the functionality for the prototype system planned for Phase III as well as the Algorithm Evaluation System used in this project.

2.5.1 Prototype System External PCE

The functional allocation of the PCE is depicted in Figure 2-8. It depicts both a basic set of functions that will be implemented in an initial prototype (shaded) as well as enhanced functions that will be implemented later (white). These functions are described in the following sections.

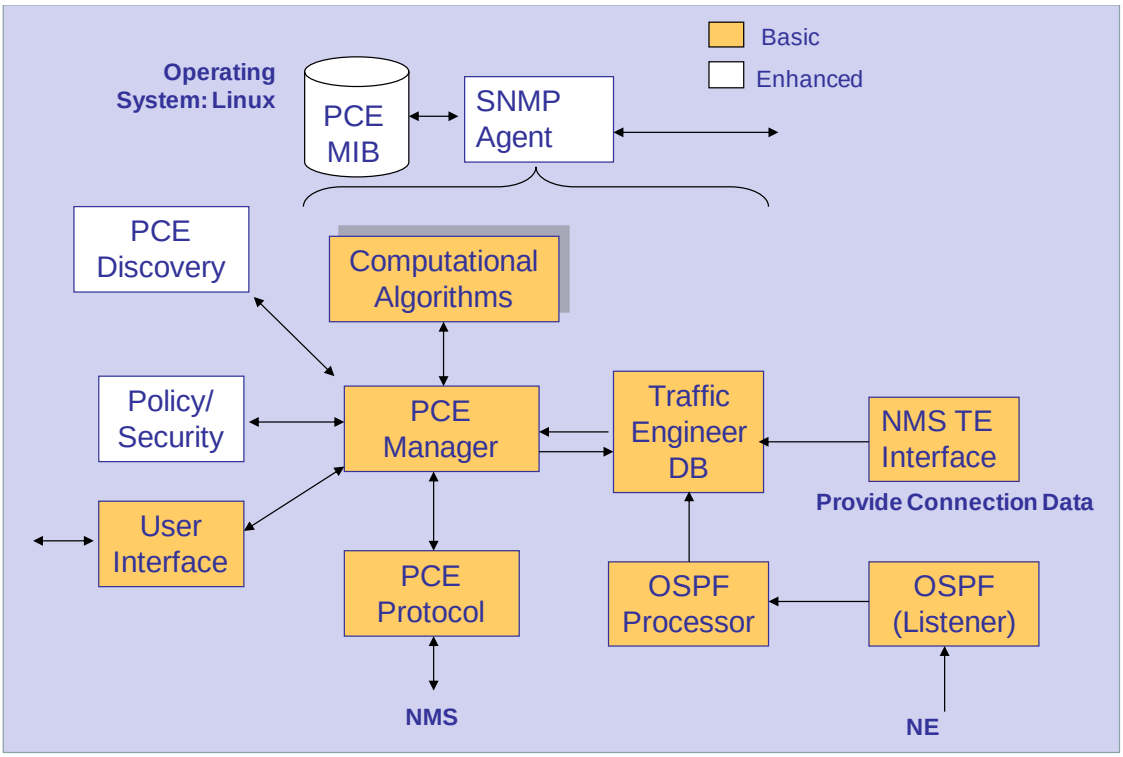

Figure 2-8: Target PCE Functional Allocation

2.5.1.1 PCE Manager

The PCE Manager performs the overall control of the PCE operation. In this role, it has three major responsibilities:

- Receive and process inputs from external sources such as the NMS, and the user; and provide responses as necessary,
- Provide traffic engineering updates to the TED (LSP set up, activate, release) and extract current traffic engineering data when needed,
- Invoke the Path Computational Algorithms.

In the future, the PCE Manager will also invoke the Policy/Security function that will control access to the PCE and co-ordinate the PCE Discovery function that will allow Path Computation client to learn the status of the PCE. It will also support SNMP management by the NMS.

In this functional allocation, the PCE Manager is GMPLS switching technology dependent while the Computational Algorithms are technology independent. Therefore, it is the responsibility of the PCE to convert the technology dependent GMPLS network into a generic graph so that the selected computational algorithm function can execute its optimization method. For example in the Lambda switching case, it is the responsibility of the PCE Manager to prune the network graph to include the selected wavelength so that the network graph provided to the algorithms includes only the selected wavelength. In generating the generic network graph, the PCE Manager will implicitly enforce the wavelength continuity constraint.

The PCE Manager will become more complex in the future. In Phase III, when hierarchical LSPs may be supported, the PCE Manager will generate graphs for each hierarchical region. For

example in this case, the PCE Manager will generate one graph corresponding to the transport network (lambda switching technology) and one network corresponding to packet switching that overlays the transport network. Also, when protected paths are generated, the PCE Manager will enforce the disjointness constraint between working and protected paths.

It is envisioned that there will be multiple Computational Algorithms implemented in the PCE so that the relative performance can be compared. The algorithm to be used is determined by user configuration. Then the PCE will do the necessary pre-processing and then invoke the requested method.

2.5.1.2 Computational Algorithms

The suite of computational algorithms implemented in the PCE will address both switched and fixed path routing (as discussed in Section 3.3). It is implemented as a generic optimization method so the same method applies to any switching technology.

2.5.1.3 Traffic Engineering Database

The Traffic Engineering Database contains the network topology, link loading, and for "stateful" operation the path of the individual LSPs. It all includes such parameters as the number of wavelength per DWDM link.

The Traffic Engineering Database (TED) is a compiled and optimized database of the network topology and resource information. It is constructed using several inputs, including input from management plane, as well as the link state information received through the distributed routing protocol, OSPF. Some simple control plane implementations choose to directly use the database of the link-state routing protocol as their topology database, but a separate topology database used in this implementation provides several advantages, including

- Faster path computation (because of compilation of data into highly accessible form, which generally reduces the search time for network elements from linear time to constant time),
- Protection against routing transients and invalid topology and resource information, and
- Allowing multiple sources of information (user, management plane) in addition to the routing protocol

2.5.1.4 User Interface

The user interface allows the user to select the specific Computational Algorithm and specify the algorithm parameters. As described in the Section 3, the objective function used in the optimization method consists of starting time, duration, and path length components. The user interface allows the user to specify the weights assigned to each of these objective function components.

In addition, the user may configure the PCE to use either switched or fixed path routing.

2.5.1.5 PCE Communications Protocol

In the PCE, the PCE Communications Protocol [23] enables the PCE to receive scheduled LSP requests from the NMS and return the computed paths. To support scheduled requests, new objects must be introduced to specify start times and durations.

For "stateful" operation, it will also be enhanced enabling the NMS to notify the PCE that certain network events have occurred, e.g., LSP path set success/fail, LSP release. Details of the enhancements are provided in Section 4.

The PCEP runs over TCP such that reliable data transfer and flow control are provided by the underlying protocol stack.

2.5.1.6 OSPF Processor

The OSPF Processor receives the standard OSPF-TE LSAs from the OSPF Protocol, parses them, and forwards them to the TED in a data structure easily processed by the PCE Manager. For stateless operates, it also parses the proprietary Lambda aware routing fields to provide the availability of wavelengths on DWDM link to the PCE.

2.5.1.7 OSPF Protocol

The OSPF Protocol executes the OSPF-TE protocol with the Lambda aware enhancements, i.e., it listens to OSPF to obtain the LSAs from a Network Elements, but does not generate any opaque LSAs with TE information. However, it does need to synchronize with the NE.

2.5.1.8 NMS TE Interface

Since the TE data is provided to the PCE using the PCEP, this interface is not needed in this project. However, in the future, it may be used to provide a web services interface between the NMS and PCE.

2.5.2 Algorithm Evaluation System (AES)

Since a major objective of this project is to analyze the performance of computational algorithms, it is only necessary to implement the PCE. Its functional architecture is depicted in Figure 2-9 and described in the following sections.

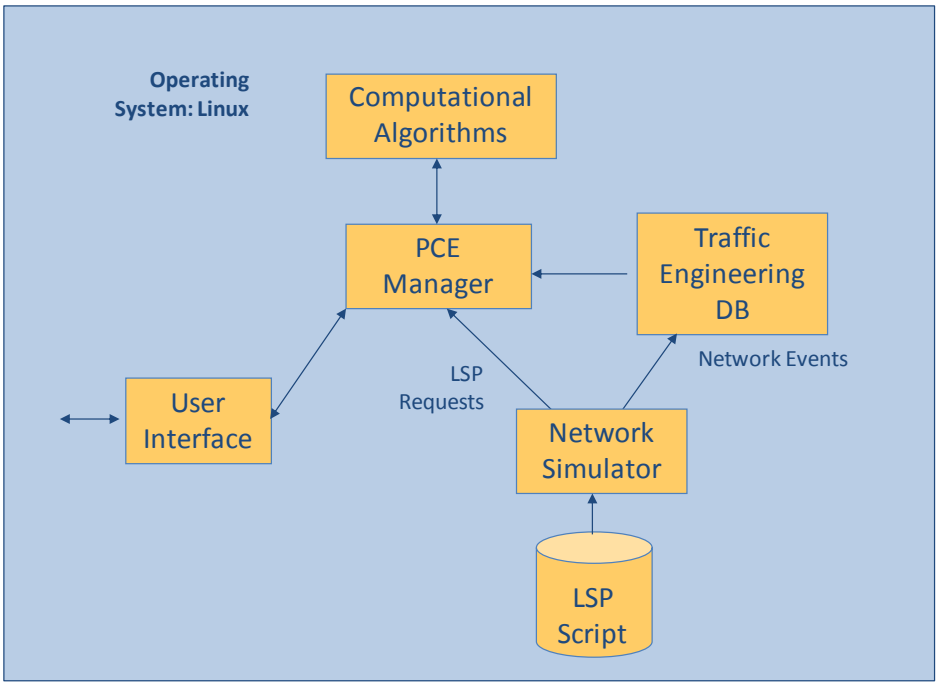

Figure 2-9: Algorithm Evaluation System

2.5.2.1 PCE Manager

Same as Phase II except it gets input from the simulator rather than PCEP or OSPF-TE. In year 1 it will support both Lambda and packet switching technologies

2.5.2.2 Computational Algorithms

The suite of Computational Algorithms in the Algorithm Evaluation System is generic and may be applied to any network graph. Therefore, the algorithms will transition to the commercial product.

2.5.2.3 Traffic Engineering Database

The TED in the Algorithm Evaluation System supports both Lambda and packet switching technologies. Otherwise, the TED is the same as described above.

2.5.2.4 User Interface

The user interface in the Algorithm Evaluation System allows the user to select either Lambda or Packet switching technologies. Otherwise, the user interface is the same as described above.

2.5.2.5 Network Simulator

The Network Simulator models the flow of LSP requests from the NMS and the network events reported to the PCE by the OSPF Listener and the NMS. The LSP requests are randomly generated. Two options are available:

- Uniform traffic distribution among all network nodes,
- Region oriented distributions where the traffic between nodes is dependent on their geographic location.

In reporting the network events, the simulator provides the data required to populate the Traffic Engineering Database (network topology, link loading, and for stateful operation the path of the individual LSPs, as well as the number of wavelength per DWDM link.

Based on the mode (stateless or stateful), the simulator operates in either a passive or active mode. For the stateless mode, the simulator operates passively in that it generates events and provides them to PCE Manager, receives response from the PCE Manager, and generates statistical data summarizing the results. For the "stateful" mode, the simulator generates additional events for the PCE Manager, e.g., indicating that an LSP has been successfully set up, LSP set up has failed, and LSP has been released.

2.5.3 Representative Network Element

We have selected our LN2000 as a representative Network Element for the Phase III prototype demonstration. Figure 2-10 depicts the functional allocation on the LN2000 System Node Manager (SNM), its major processing element. While the LN2000 is very complex, enhancements to support scheduled services are required only in selected functions as indicated by the shading. These modifications are described in the sections below.

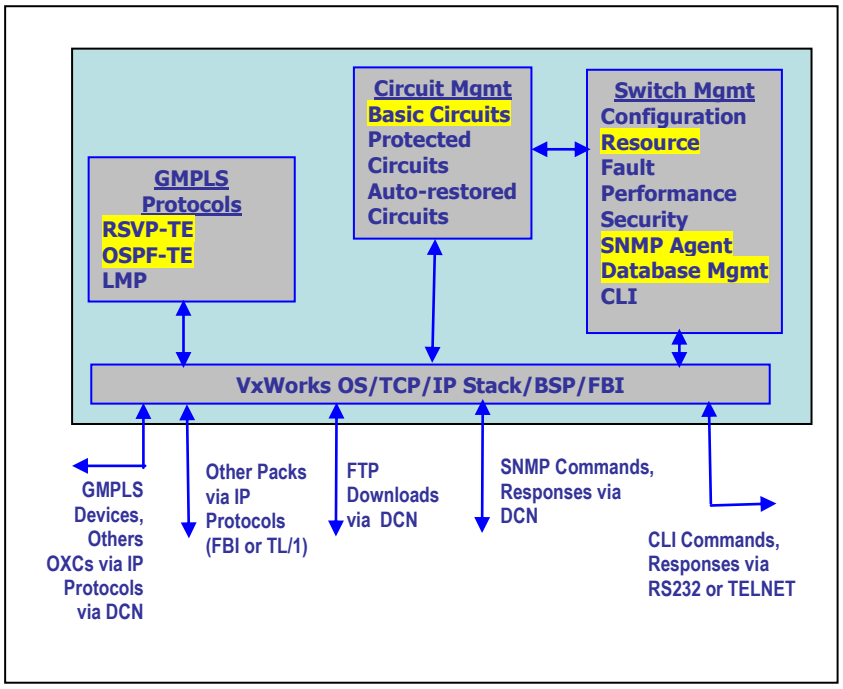

Figure 2-10: Network Element Functional Allocation (SNM only)

2.5.3.1 Circuit Management

The Basic (Unprotected) Circuit will be modified to support Scheduled Circuits where the start time and duration will be specified in the SNMP request initiating the set up of the circuit.

2.5.3.2 GMPLS Protocols

The signaling and routing protocols require modification to support scheduled LSPs as well as on demand LSPs. Nodes with the enhanced protocol suite are interoperable with GMPLS nodes that are not. The source node of the scheduled LSP must support the enhancements, but transit nodes may not. In this case, the transit nodes establish the service immediately by setting the selected cross-connects. More detail on interoperability is presented in Section 4 that addresses the extensions to the GMPLS protocols.

For fixed path services, there is one LSP supporting the service. For switched path services, there is a separate LSP for each different path that is used during the service duration. Refer to Section 4 on the process for switching from path to path.

2.5.3.3 Switch Management

For the set of Switch Management functions, the primary change is the Resource Manager that now must be time aware. The Resource Manager (RM) is the software module in charge of local switching and resource allocation; it provides an abstraction layer on top of hardware, making the control plane software more modular and hardware-independent. Resource manager must be *persistent*, preserving the local resource information across node or control plane reboots. To support distributed scheduling, resource manager must store and keep track of resource utilization over time, and therefore hast to keep track of real time, through both an on-board *real time clock*, as well as network interface to a centralized time server, through *Network Time Protocol (NTP)* or *Simple Network Time Protocol (SNTP)* [29]. It is assumed that resource manager modules in all nodes are synchronized with a predictable and known accuracy, i.e., the

time difference between any two resource managers in the scheduling domain does not exceed a known limit.

Also, the SNMP Agent must now support time parameters specifying the start time and duration. In addition, for switched path services, the agent must be aware of the path and timing for each LSP used in the service.

2.5.3.4 VxWorks OS/Protocol Stack

The commercial operating system and protocol stack are used. There are no changes.

2.5.4 Representative NMS

In this project, the key functions of the NMS involve Connection Management. As shown in Figure 2-11 to support scheduled LSPs, the NMS includes of the T(imed) Connection Manager providing control of the connection functions, a PCE Client to obtain paths from the PCE and an SNMP Manager to instantiate the circuits. While a GUI is depicted in the figure to allow the user to enter circuit requests manually, a script may be used such that circuit request are entered electronically.

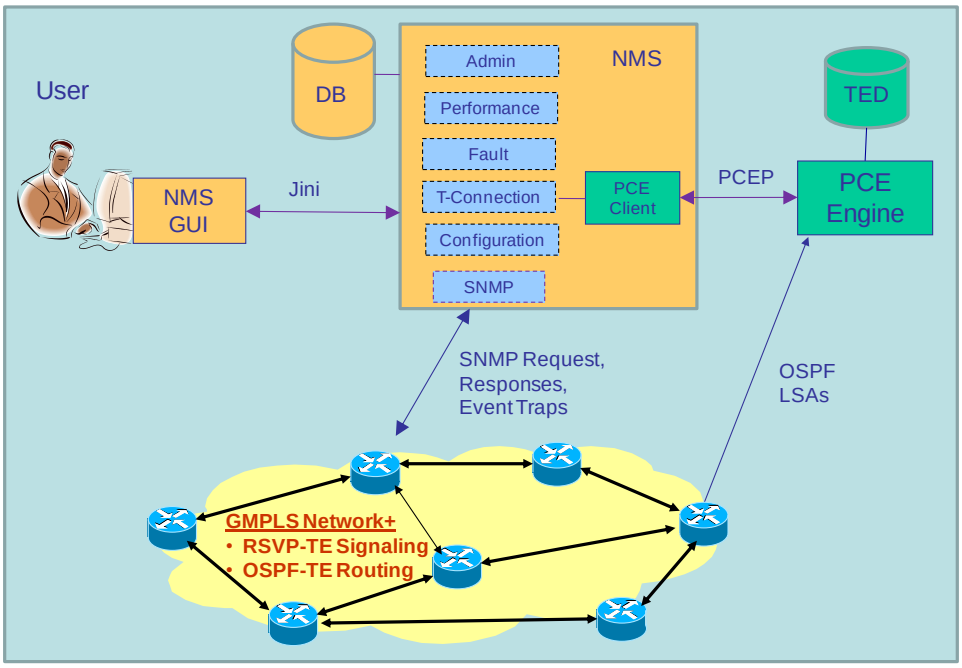

Figure 2-11: NMS Functionality – PCC Resident in NMS

Note the implementation details of the NMS will be decided during the development phase. It may be enhancement of the Lambda SDS implemented in Java or a new standalone component implemented in C or C++.

If the PCE is implemented in the NE, the NMS functionally still requires the Timed-Connection Manager. Figure 2-12 depicts this functionality.

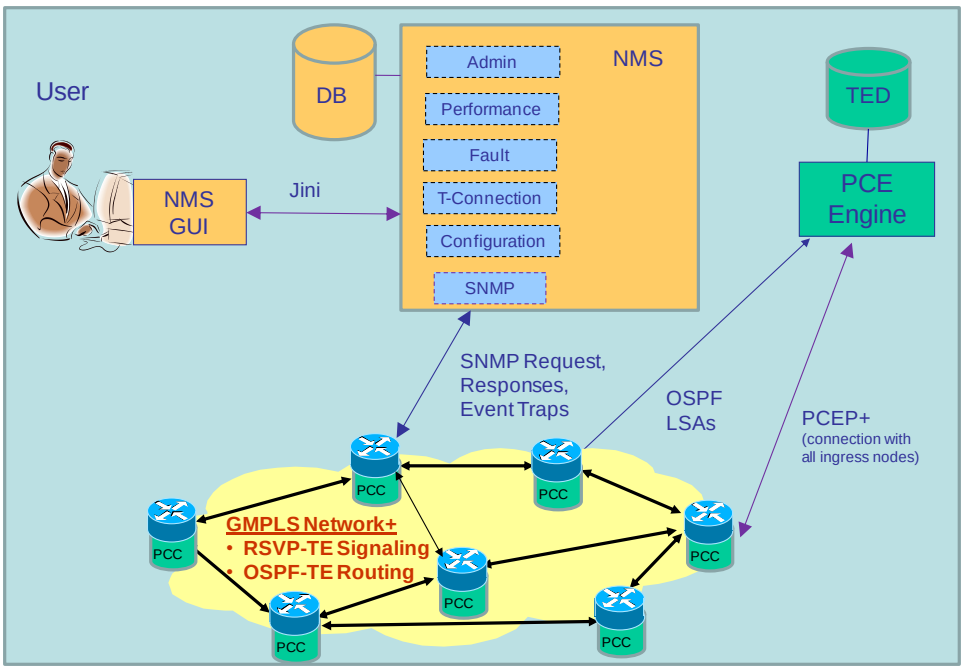

Figure 2-12: NMS Functionality – PCC Resident in NE

2.5.4.1 Connection Manager

The Connection Manager accepts LSP requests either from the GUI or a script, and then it initiates:

- path computation by sending a request to the PCE using the PCE client,
- path set up by sending an SNMP command with a GMPLS Explicit Route Object (ERO) to the source NE.

When LSP has been completed (either successfully or failed), the NMS updates the database. For stateful operation, it forwards these updates to the PCE.

2.5.4.2 PCE Client

The PCE Client implements the PCE Communications Protocol [23] enabling the NMS to access the PCE. This protocol runs over a TCP stack.

2.5.4.3 SNMP Manager

The SNMP Manager enables the NMS to send commands to the NE and receive traps from the NEs. It will be enhanced to include the LSP time dependent parameters.

3 ALGORITHMS

The purpose of this section is to specify the algorithms, parameters, and user options that will be implemented to support distributed scheduling using GMPLS. The content of this section includes the shortest path algorithms, optimization algorithms based on desired start time and duration, and control methods.

Section 3.1 describes the overall algorithmic architecture that includes the PCE Manager, PCE Optimizer, and PCE Path Selection (Shortest Path) modules. Then Sections 3.2 to 3.4 describe the PCE Manager, PCE Optimizer, and Path Selection.

3.1 Algorithm Architecture

In Phase I, we developed a suite of algorithms to support GMPLS distributed scheduling so the PCE will implement a set of options supporting multiple modes and corresponding algorithm parameters. Figure 3-1 depicts the algorithm taxonomy in detail. First, it illustrates that the algorithms may be operated using either threshold driven or on demand network updates. Second, the algorithms may use either switched or fixed paths depending upon whether the network path of the service is allowed to change during the service duration.

In addition, the PCE may be configured to optimize start time, path length, or duration or a combination. As part of the optimization, shortest path algorithms are required.

Although not depicted, the algorithms may also be "stateless" or "stateful" depending upon whether the PCE maintains the status of individual LSPs. As addressed in Section 2, the algorithms will be implemented in a "stateful" mode to facilitate the enforcement of the wavelength continuity constraint in all-optical networks. In the future the algorithms will be implemented in a stateless mode to support other switching technologies.

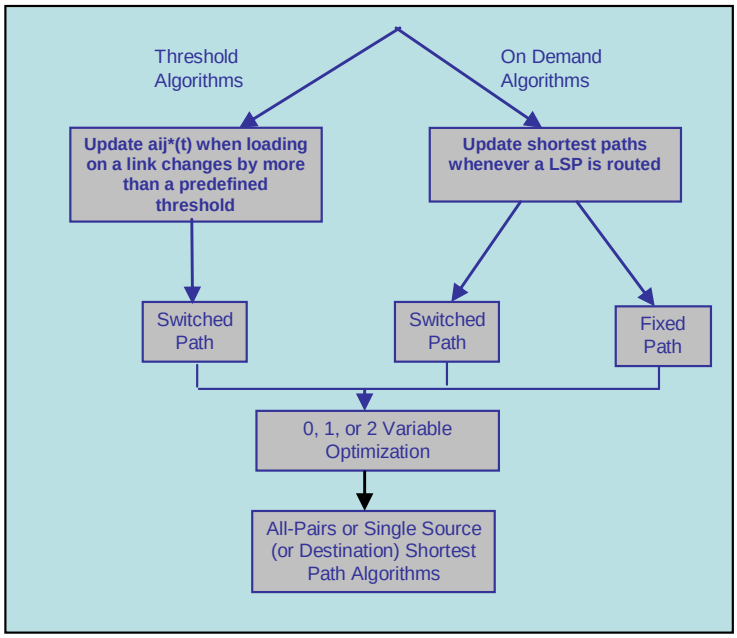

Figure 3-1: Algorithm Architecture

In this section, the terminology jump points and solution points is used. Jump points refer to the times where bandwidth availability changes while solution points refer to the number of candidate solutions considered in the optimization algorithm. There are typically many more solution points than jump points. Section 3.3 describes these concepts in more detail.

Fixed path algorithms require shortest path calculations per solution point rather than per jump point. Therefore, it is not possible to pre-store the shortest path values because solution

points are unique to the service request. As a result, threshold algorithms are not applicable to fixed path algorithms.

As depicted in Figure 3-2, the algorithmic related modules of the PCE consist of:

- PCE Manager that provides overall control,
- PCE Shortest that computes a shortest path, and
- PCE Optimizer that determines the optimal path.

As indicated in the figure, the PCE Manager is technology (Lambda, Packet, etc.) while the other modules are technology independent. For example, the PCE Manager is aware that that the underlying network employs Lambda switching and provides a generic network of nodes and links to the PCE Shortest Path and PCE Optimizer.

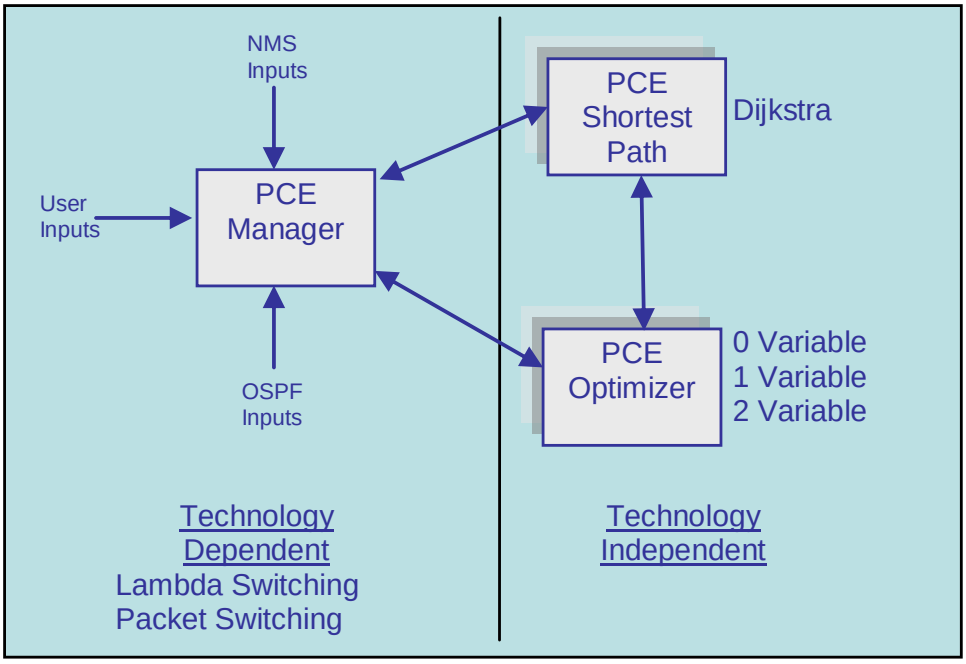

Figure 3-2: Algorithm Functional Architecture

For shortest path computation, it is planned to implement only Dijkstra because it is the only practical option for fixed path routing. In the future, a set of algorithms, e.g., Bellman-Ford, Floyd-Warshall, may be implemented to optimize switch path routing. These algorithms compute shortest path for a single or destination as well as shortest path between all nodes.

For fixed path routing, the PCE Manager invokes the PCE Shortest path module and provides the shortest path distance when it invokes the PCE Optimizer. Since fixed path routing is more complex and requires a larger number of shortest path calculations, the PCE Manager provides the time based generic network to the PCE Optimizer. Then the PCE Optimizer performs the starting and duration optimization; it will invoke the PCE Shortest Path module if the shortest path values are not provided by the PCE Manager (fixed path routing).

The PCE Optimizer uses the generic network and the shortest results to perform an optimization. The optimization may be:

- Zero variable where start time, ts, and duration, td, must be satisfied exactly,
- One variable optimization where either the start time, ts, or the duration, td, is optimized as the primary objective,
- Two-variable optimization where both ts and td are optimized.

When one variable optimization is performed, the starting time, ts (or td), is first optimized as the primary variable. Then a secondary optimization may be performed on the other parameter duration, td (or ts) and path length. In two-variable optimization, the path length and starting time are both optimized concurrently.

The current suite of algorithms determines an optimal schedule. However, the protocols suite is sufficiently general to support a schedule set. Enhancement to generate a schedule set appears straightforward but is a future activity.

3.2 Control – PCE Manager

3.2.1 Overview

As described in the System Architecture section, the PCE Manager accepts LSP requests from the NMS/PCC and invokes the PCE Shortest Path and PCE Optimizer. It also receives LSP state updates from the NMS/PCC. For study purposes, it also accepts the following configuration options from the user specifying the use of:

- Switched or fixed paths,
- Threshold or on demand updates,
- Zero, one, or two variable optimization,
- Dedicated bandwidth for scheduled services or shared bandwidth used by scheduled and on demand services.

In addition, the PCE Manager accepts algorithm parameters, e.g., weighting parameters indicating the relative importance of starting time, duration, and path length in two variable optimization.

The overall framework developed in this project supports both Scheduled and On Demand (traditional GMPLS) services. The major impact on supporting both types of services affects bandwidth allocation. This issue is addressed in Section 3.2.5. Since On Demand Services are the same as traditional GMPLS services, they are not modeled in this project.

3.2.2 Network Graph Formulation

3.2.2.1 Lambda Formulation (for all optical switching)

The fundamental path scheduling problem is that given desired start-up time Ts and duration Td, find a path from source node i to destination node j, that minimizes some objective function ϕ of the selected start time, ts, and duration, td, within an allowable time period (Tmin, Tmax). For this optimization to be performed, it is necessary to be provided a network with known available capacity.

For an optical network, the path is specified in terms of the starting node IP address, and egress interface, a set of intermediate nodes with ingress and egress interfaces, and destination

node with ingress interface. The scope of this work is to find a path from the source node to the destination node. Details concerning dropping the path to a particular egress port are not addressed in this study. This is handled by the network protocols.

It is also assumed that all requests for scheduled services have the same priority. Therefore, GMPLS priorities and pre-emption need not be supported for scheduled services.

This work is applicable to both uni-directional and bi-directional paths. If the path is bidirectional, it is required that bandwidth be available in both directions so the path in both directions to follow the same set of nodes. However, the bandwidth requirements may not be equal in both directions.

The major complexity in scheduling is that the available bandwidth is changing over time based on network updates, in this case, wavelengths. Figure 3-3 depicts a simple case corresponding to one wavelength where all of the links are available at the outset [t0, t1].

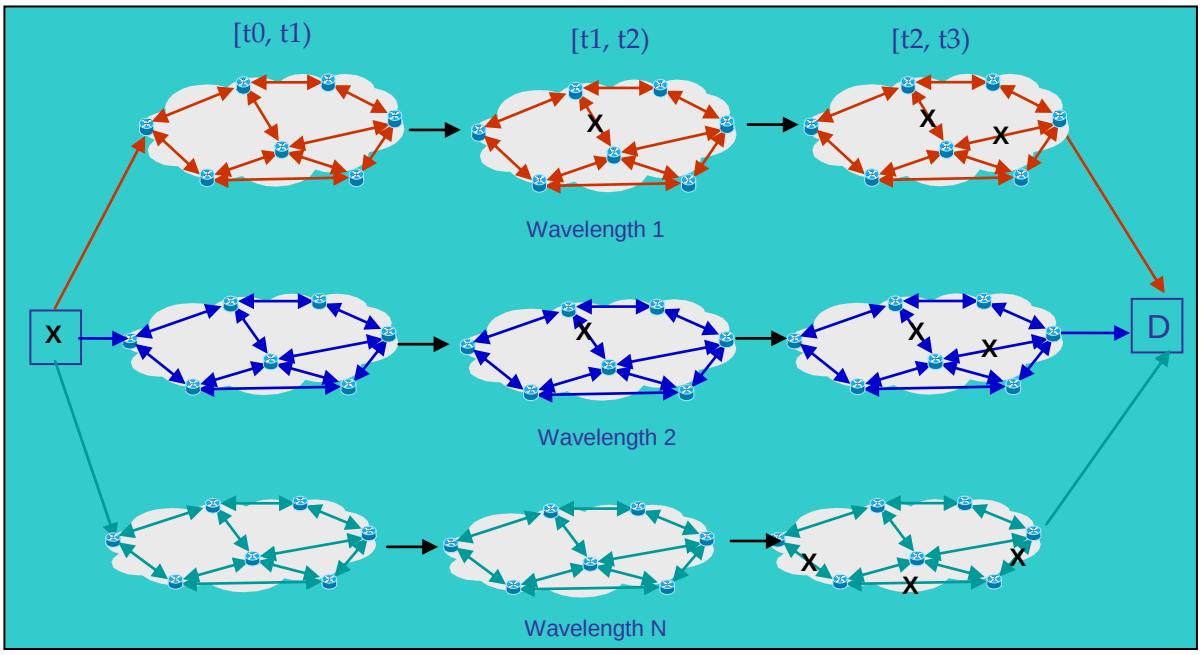

Figure 3-3: Time Dependent Network Capacity

Then for the time intervals [t1, t2] and [t2, t3], some of the links become available because the wavelengths have been assigned to LSPs. The time instances where the available capacity changes are referred to as jump points, i.e., t1, and t2 in the figure. Note, the convention in defining time intervals is that the left end is closed (inclusive) and the right end open (non-inclusive).

In this project for all-optical networks, it is assumed that the endpoint transponders are tunable. Enhancement to support fixed wavelength transponders is straightforward because only a subset of the tunable solutions must be considered.

For this work, the following general objective function ϕ defined in terms of ts and td is utilized:

$$\emptyset(t_s, t_d) = \alpha|T_s - t_s| + \beta|T_d - t_d| + \frac{\gamma}{u_{ij}}\frac{}{t_d}\int_{t_s}^{ts+td} a *_{ij} (t)dt \text{ where}$$

a_{ij}*(t) is the shortest distance between i and j at time t and

u_{ij} is a normalizing parameter equal to maximum of a_{ij}*(t) over the (Tmin, Tmax).

By setting weighting parameters, α, β, and γ, one can define the relevant zero, one or two variable optimization problem. Note as addressed below (Section 3.3.3), the objective function for fixed path optimization is slightly different.

Note the objective function uses absolute differences in start time and duration. The use of relative differences was considered using the term:

$$| 1 - ts/Ts |.$$

However, the relative error would not be stationary. For example, a 5 time unit difference would be much larger for Ts equal to 100 (1 – 95/100 = 1/20) than for Ts equal to 1000 (1-995/1000= 5/200). Thus, the absolute difference approach is used.

In this project, the scheduling is also done on dimensionless grid points. For future application, the grid may be parameterized. For example the default grid interval is 15 minutes where services begin and end at X:00, X:15, X:30, and X:45, i.e., on the hour, quarter past the hour, half past the hour, and ¾ of an hour past the hour.

For optical networks that use all optical switching, the PCE manager is responsible for enforcing the wavelength continuity constraint. If the network uses fixed wavelength transponders, the PCE prunes the network according the availability of the selected wavelength. For fixed wavelength transponders or tunable transponders, it is assumed that the allowable wavelength is determined from the LSP request.

3.2.2.2 Packet Formulation

The packet formulation of the path routing is very similar to that of the lambda switching formulation described above with minor differences – involving mostly simplifications. First, with packet switching as shown in Figure 3-4, only a single layer network must be addressed as opposed to a layered network of wavelengths described above for Lambda switching. In this formulation, a link is included in network if capacity exists to support the LSP being assigned.

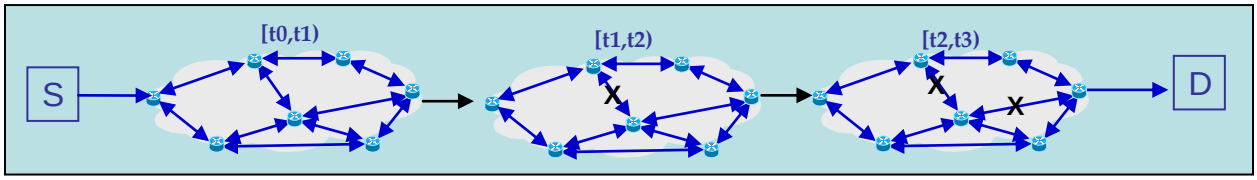

Figure 3-4: Time Dependent Network Capacity

Second, instead of selecting a data rate from a set of wavelengths, the user specifies a data rate in terms of bps as specified in the PCE Communication Protocol (PCEP) RFC [23].

3.2.3 Routing

Since the available bandwidth is changing over time, the shortest path between nodes i and j may also change during this time. Two cases, switched path and fixed path routing, are addressed in the following sections.

3.2.3.1 *Switched Path Routing*

In switched path routing, the path supporting the service may change during the service duration. In this case, the PCE Manager generates the network graph for each time interval where the available bandwidth changes. For the network with bandwidth availability changing at t1, t2, t3, t4, and t5, Figure 3-5 depicts the path assignments for an LSP beginning service t2. As shown in the figure, the service will have three paths corresponding to the time intervals [t2,t3], t3,t4), and [t4, ts+td]. Thus at time t2 and t3, it is necessary to switch paths. It is planned to use a "make before break" protocol to perform this switchover. Details of "make before break" are presented in the Protocols section below.

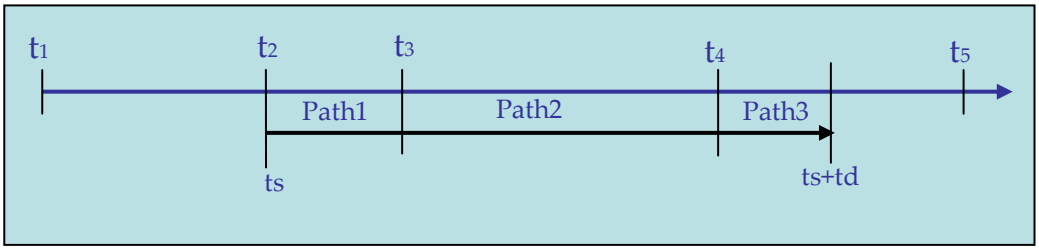

Figure 3-5: Switched Path Routing

Let a*ij(t) become the cost of the shortest path from node i to j at time t. Then as shown in the figure,

Path1 is the shortest path during (ts, t2),
Path 2 is the shortest path during (t3, t4),
Path 3 is the shortest path during (t4, ts+td).

The advantages of switch path routing are that 1.) LSP blocking will be reduced and 2.) the network links may be used more efficiently.

3.2.3.2 *Fixed Path Routing*

In fixed path routing, the path must stay the same for the duration of the service as depicted in Figure 3-6. In this case, the PCE Manager must prune the network such that the links used for the shortest path calculation have sufficient capacity to support the service for the full service duration, [ts , ts+td].

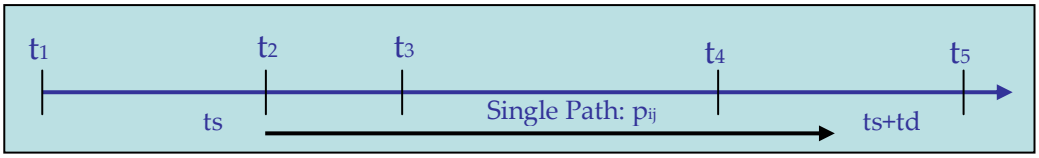

Figure 3-6: Fixed Path Routing

For fixed paths, let c(pij)(t) equal the cost of path i to j at time t. It is assumed that this cost is constant during the service interval (ts, ts+td). When performing fixed path optimization, it is not necessary (or possible) that pij be the shortest path during the service interval (ts, ts+td), but it must only be feasible during the interval.

The advantages of fixed path routing are software and operational simplicity. There will be no disruptions due to switching paths as may occur for switched path routing.

3.2.4 Stateless and Stateless Operation

The initial implementation of the PCE will be a stateful operation where it maintains the results of the path computation, provisionally updates the network loading in the TED, and receives LSP connection status from the NMS to update the LSP status in the TED. In this case, the PCE does not need the OSPF bandwidth updates.

The advantages of a stateful include:

- enforcing the wavelength continuity constraints,
- handling back-to-back requests due to heavy user load,
- supporting back to back auto-restoration requests due to network failures,
- performing pre-emption and re-optimization of LSPs.

However, stateful operation is more complex because of the complexity of maintaining consistent data as LSPs set ups are attempted and fail as well LSP releases occur.

In the future, stateless operation may be supported where upon request, the PCE computes a path for the NMS upon request and returns, but it does not store the path or update network loading based on the path. The PCE waits for OSPF to update the link loading and does not maintain any LSP specific information.

3.2.5 Bandwidth Control

This section describes two approaches for updating the shortest path calculations, updating shortest path on receipt of the service request and update based on exceeding link utilizations. This section describes both approaches and presents example scenarios.

In addition, two approaches are supported for bandwidth availability:

Partitioned – bandwidth allocated individually to scheduled and on demand services,

Shared – bandwidth is shared by scheduled and on demand services.

The updating approaches presented below are applicable to both of the above options. As mentioned above, scheduled services are modeled in this work. However, the Protocols Section addresses the enhancements necessary to support both services.

3.2.5.1 Bandwidth Updates on Request

When the updates are performed on demand, the shortest path calculation is performed whenever a new LSP service request is being handled. As depicted in Figure 3-7, the network updates are provided as they occur. However, in this case the PCE Manager does not invoke the PCE Shortest Path module to update the shortest path $a_{ij}^*(t)$ (in the case of switched paths) until a service request is received from the NMS.

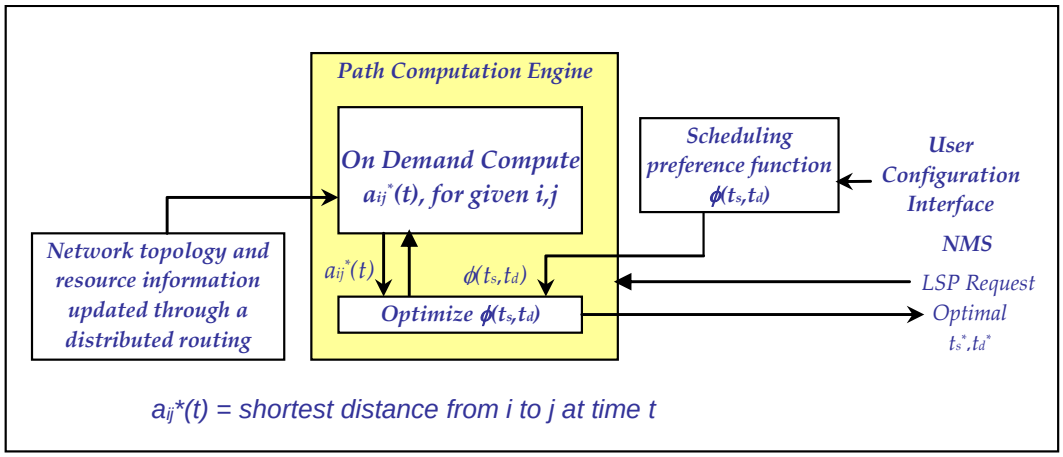

Figure 3-7: Flow with Updates on Service Request

When the PCE is configured, the user must:

- specify the type of optimization to be performed (zero, one, or two variable) and
- provide algorithm parameters as described below.

Figure 3-8 depicts the message flow for servicing an LSP request when the shortest path is updated On Request for Switched Paths.

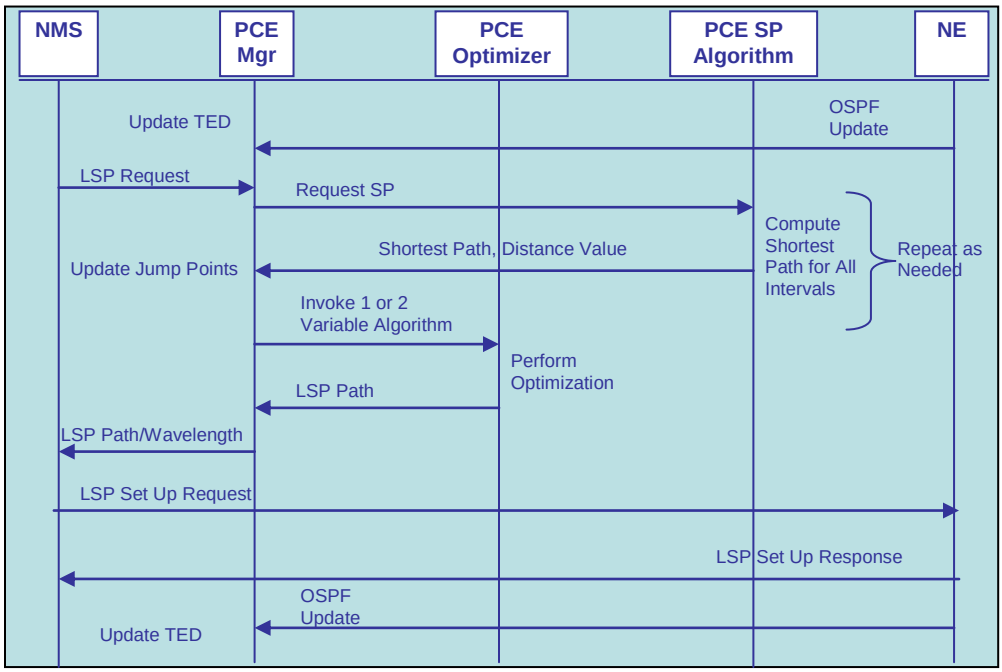

Figure 3-8: On Request Shortest Path Update Scenario – Switched Paths for Networked Implementation

As shown in this case, the shortest path is updated when the LSP request is received. Note that the shortest path is updated for each time interval relevant to the service request. Therefore, it is likely that the PCE Shortest Path calculation will be invoked many times.

While example above is depicted for switched path routing (computing shortest path aij*(t)for all jump points), the same concepts may be applied for fixed path routing using c(pij)(t).

However because of the shortest paths required for each solution point, the PCE Manager provides the PCE Optimizer with the time based network topology and the PCE Optimizer invokes the shortest path calculations. Figure 3-9 depicts the modified scenario for fixed path routing.

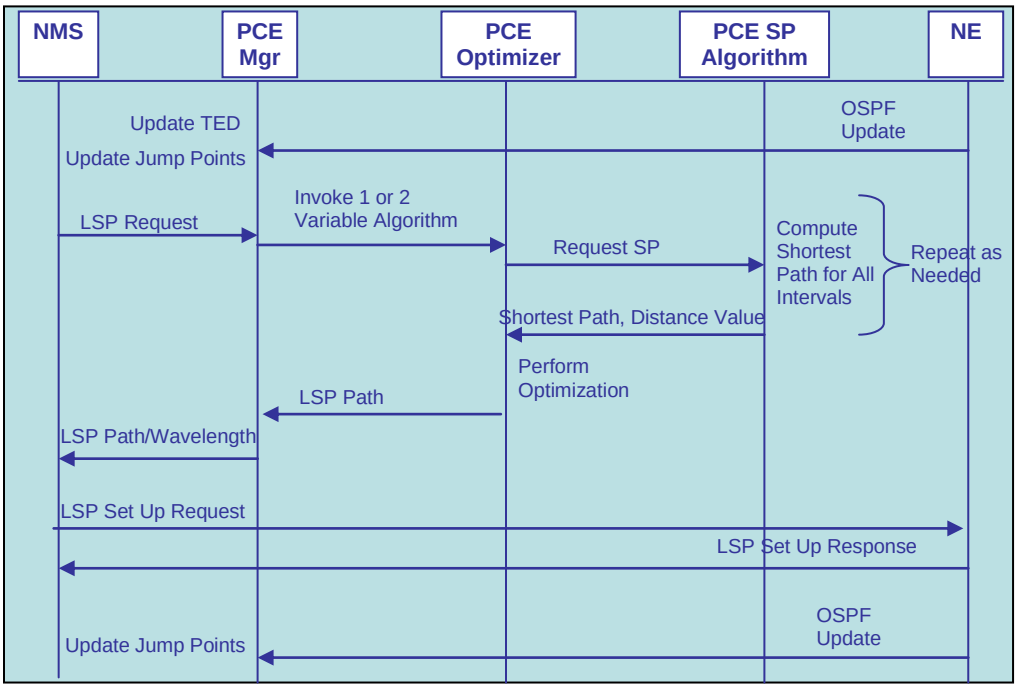

Figure 3-9: On Request Shortest Path Update Scenario – Fixed Paths

3.2.5.2 Threshold Bandwidth Updates

When the updates are based on a threshold, the PCE Manager invokes an additional algorithm to compute changes in link utilization and compare the changes to a configured threshold. Figure 3-10 depicts the logic for this approach. Note this approach applies to switched path routing only because of complexity as discussed in Section 3.1.

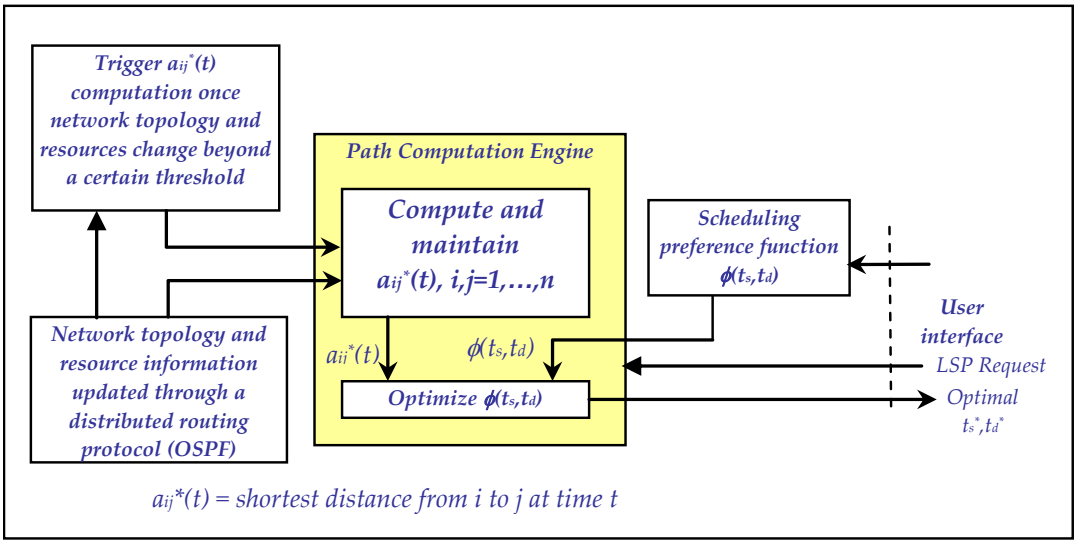

Figure 3-10: Flow with Threshold Updates

If the change in link utilization, δU, exceeds the allowable threshold, U_L, over any time interval for any link, the PCE Manager invokes the PCE Shortest path to update $a_{ij}^*(t)$ for the relevant time intervals. When a new LSP service request is being handled, the PCE Manager retrieves the $a_{ij}^*(t)$. It is not necessary to perform any shortest path calculation.

For some network operations, there may be many updates that do not affect bandwidth availability, e.g., adding only a low bit rate packet switched LSP using 10G TE links. Thus, this approach may reduce the shortest path computing load.

The reduction must be significant to be computationally efficient because the threshold algorithm performs an all pairs computation while the on request update only does a single source -destination computation. In the case of threshold updates, a variation is to identify the i,j pairs that are affected by the bandwidth change and only update those pairs – using a traditional technique.

Figure 3-11 depicts the message flow for servicing an LSP request when the shortest path is updated based on utilization thresholds. As shown in this case, the PCE Manager checks the thresholds whenever an OSPF update is received. If the threshold is exceeded for some link during a time interval, then the PCE Manager will invoke the PCE Shortest Path module to update $a_{ij}^*(t)$ for that time interval. Note that the shortest path is updated for each time interval where the threshold has been exceeded. Therefore, it is likely that the PCE Shortest Path calculation will be invoked multiple times.

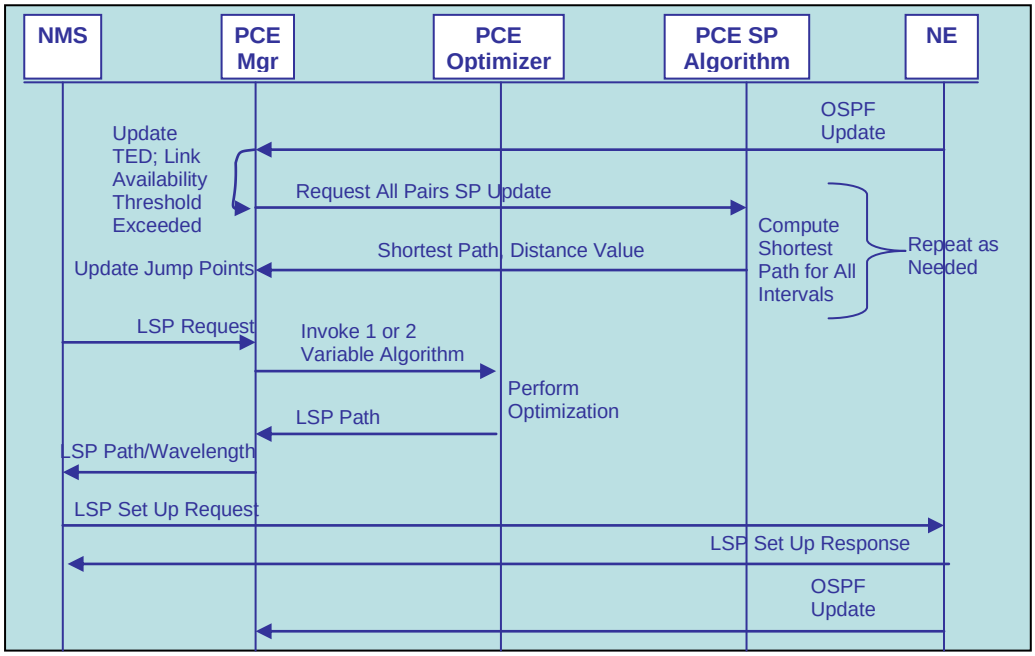

Figure 3-11: Threshold Update Scenario for Switched Paths

As mentioned above, this approach is not applicable to fixed path routing because of the number of shortest path calculations required, i.e., per solution point.

There are two methods for implementing threshold algorithms based on handling the LSP request that caused the threshold to be exceeded.

Soft Threshold (depicted above) - when service request k required in BW b causes the threshold to be exceeded, capacity is still assigned to service request k using the current shortest path information. Then the links whose threshold has been exceeded are deleted from the network graph and the shortest paths are recomputed for use in servicing future requests.

Hard Threshold – when service request k required in BW b causes the threshold to be exceeded, the links whose threshold has been exceeded are deleted from the network graph and the shortest paths are recomputed. Then capacity is assigned to service request k based on the new paths.

In the "hard" algorithm, the utilization threshold is never exceeded while in the "soft" algorithm there will be slight overloading on some TE links.

3.2.6 Algorithms Interface

This section describes the PCE Manager interface with the PCE Shortest Path and PCE Optimizer for both switched path routing and fixed path routing. In both cases, it is necessary for the PCE Manager to prune the technology dependent network to a technology dependent network. The pruning of the network, the switched path interface, and fixed path interface are described in the following sections.

Details of the data structure representing the generic network are not addressed in this document. These details are left to the software design, e.g., may use either an incidence matrix or list of arcs to represent a network.

3.2.6.1 Network Pruning

The PCE Manager provides a generic directed network of nodes, links and link costs for use by the PCE Shortest Path module. Since the PCE Shortest Path and PCE Optimizer are technology independent, the PCE Manager performs the selection of the wavelength and forwards only the single wavelength network to the PCE Shortest Path or PCE Optimizer. For example for the Lambda network shown in Figure 3-3, the PCE Manager may pass only the blue network corresponding to wavelength 2.

If wavelength conversion is permitted, the PCE Manager would provide a generic network with multiple wavelengths (as depicted above in Figure 3-3) and with links between the networks showing where wavelength conversion is provided. However, wavelength conversion is a Phase III activity.

If tunable wavelength transponders are being used, the PCE Manager will evaluate all wavelengths and select the best candidate as the solution. Alternative approaches may be used such as first fit. In this option, As soon as a feasible wavelength is found, use the best path for that wavelength.

In the case of packet switching discussed above, the formulation of the generic network is much simpler because a link is available if and only if there is available capacity. As a result, similar problems do not arise because there is only one network layer to consider.

3.2.6.2 Switched Path Routing

3.2.6.2.1 PCE Shortest Path

For switched path routing, the PCE Manager provides the generic network to the Shortest Path module with the request to compute the shortest path between a pair of nodes, i,j or between all pairs for a particular time interval. In response, the PCE Manager receives the shortest path value(s) and corresponding path(s) in return.

To find the shortest path over all time intervals with different link bandwidth availability, the PCE Manager will invoke the PCE Shortest Path module multiple times – once for each interval. Note the PCE Manager may invoke the PCE Module on receipt of a service request or upon exceeding a change in link utilization threshold as described above.

3.2.6.2.2 PCE Optimizer

The PCE Manager provides an extensive set of input parameters to the PCE Optimizer as enumerated in Table 3-1. Note for switched path routing, the PCE Optimizer only needs the shortest path distance, not the actual path. In return, the PCE Manager receives the optimal starting time, ts*, and optimal duration, td*. With the optimal starting time and duration, the PCE Manager can derive the optimal path for switched path routing based on the shortest path results.

Table 3-1: Switched Path Optimization Parameters

Parameter	Options
Source-Destination	i, j
Optimization Type	0, 1, or 2 Varable; for 1 variable routing whther starting time or duration is the primary objective function.
Shortest Path Value	a_{ij}*(t)at the jump points
Desired starting time and duration	T_s, T_d
Earliest Starting Time Latest End Time	T_{min} T_{max}
Minimum Duration	T_{dmin}
Objective function weights (dependent on optimization type)	α = starting time weight β = duration weight γ = shortest path weight

3.2.6.3 Fixed Path Routing

3.2.6.3.1 PCE Shortest Path

For fixed path routing, the PCE Manager does not invoke the PCE Shortest Path module. It is invoked by the PCE Optimizer because of the larger number of executions.

3.2.6.3.2 PCE Optimizer

The PCE Manager provides an extensive set of input parameters to the PCE Optimizer as enumerated in Table 3-2. Since the PCE Manager does not invoke the PCE Shortest Path module, it provides the time dependent generic network to the PCE Optimizer rather than the

Shortest Path Values. In return, the PCE Manager receives the fixed path and associated cost in addition to the optimal starting time, Ts*, and optimal duration Td*.

Table 3-2: Fixed Path Optimization Parameters

Parameter	Options
Source-Destination	i, j
Optimization Type	0, 1, or 2 Varable; for 1 variable routing whther starting time or duration is the primary objective function.
Time Dependent Generic Network	Nodes, links, link distances for each time interval
Desired starting time and duration	T_s, T_d
Earliest Starting Time Latest End Time Minimum Duration	T_{min} T_{max} T_{dmin}
Objective function weights (dependent on optimization type)	α = starting time weight β = duration weight γ = shortest path weight

3.3 Optimization – PCE Optimizer

The PCE Optimizer operates on a technology independent network graph provide by the PCE Manager. This section addresses Optimization for Switched Paths and Optimization for Fixed Paths in Sections 3.3.1 and 3.3.2, respectively..

3.3.1 Switched Path Optimization

This section presents the switched path algorithms for zero, one, and two-variable optimization. For one variable optimization, timeliest path first or fittest path first optimization may be used.

3.3.1.1 Zero Variable Optimization

In zero variable optimization, the desired starting time, ts, and desired duration, td, must be satisfied exactly. Therefore, the PCE Optimizer checks if $a_{ij}^*(t)$ is finite in the interval [ts, ts+td]. If the value is finite, then the optimal path is the corresponding path(s) determined in computing $a_{ij}^*(t)$ and is returned to the PCE Manager.

Otherwise, the request is rejected as infeasible.

3.3.1.2 One Variable Optimization

3.3.1.2.1 Timeliest Path First

In one variable optimization with timeliest path first optimization, the PCE Optimizer first determines the path with the starting time, ts, closest to the desired, Ts within the range (Tmin, Tmax).

$$\emptyset(t_s) = |T_s - t_s|$$

It then optimizes a secondary objective function based on the desired duration, Td.

$$\emptyset(t_d) = \beta|T_d - t_d| + \frac{\gamma}{u_{ij}}\frac{}{t_d}\int_{t_s*}^{ts*+td} a*_{ij}(t)dt$$

where

$a_{ij}^*(t)$ is minimum cost function to reach node *j* from node *i* at time where the selected path may change at each breakpoint,

u_{ij} is a normalizing constant equal to maximum $a_{ij}^*(t)$ over the range (Tmin,Tmax),

Weighting parameter, β, γ,

Jump points t1 < t2 < t3 … < tk over the range (Tmin,Tmax).

To find the timeliest path, perform the following steps such that the primary objective function is minimized:

Order the jump points $\{t_{oi}\}$ such that $|Ts - t_{o1}| < |Ts - t_{o2}| < |Ts - t_{o3}| < |Ts - t_{o4}| \ldots < |Ts - t_{ok}|$.

As the jump points are being ordered, evaluate $a_{ij}(ts)$ at ts = Ts, t_{o1}, t_{o2}, …, t_{ok}, and pick ts* as the first value of ts such that $a_{ij}(ts)$ is finite.

Since jump points less than and greater than Ts are being considered, the strict inequality in 1 may not hold. Therefore, two jump points, upper and lower Ts, may occur. The secondary objective function may be used to break ties.

To find the fittest timeliest path, apply Theorem 2 in the Phase I Final Report:

Theorem 2: For piecewise-constant $a_{ij}^*(t)$, the optimal duration t_d^* that minimizes the secondary objective function is equal to T_d, or $a_{ij}^*(t)$ jumps at $t_s^*+t_d^*$.

This optimization is implemented by performing the following steps:

1. Compute u_{ij} equal to maximum $a_{ij}^*(t)$ over the range (Tmin,Tmax).
2. Determine the ordered jump points, $\{t_j\}$, in the interval $(t_{s*}, t_{s*}+T_d) \Rightarrow t_{j1} < t_{j2} < t_{j3}$
3. Evaluate $\phi(td)$ at each jump point {tj} and for Td stopping when $a_{ij}^*(td)$ is infinite for some evaluation point.
4. From the values computed in step 3, select td* as the value of td that minimizes $\phi(td)$.

It may happen that there are two values of ts* that optimize the primary objective function, i.e., upper and lower start times. If so, repeat the above optimization for the secondary objective function using both upper and lower values of ts* and select the solution with the smaller secondary objective function as the solution.

Figure 3-12 depicts an example showing the solution points for t_3, t_4, and $t_{s*}+T_d$.

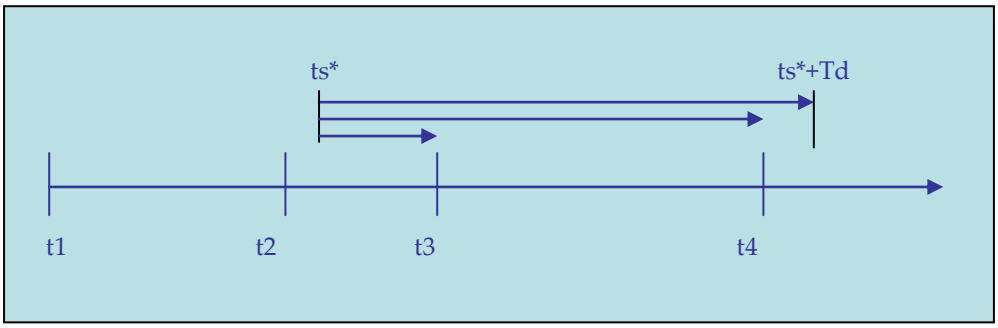

Figure 3-12: Fittest Timeliest Path Example

3.3.1.2.2 Fittest Path First

In one variable optimization with fittest path optimization, the PCE Optimizer first determines the path with the optimal duration, td*, closest to the desired duration, Td within the range (Tmin, Tmax).

$$\emptyset(t_d\ ) = |T_d - t_d|$$

It then optimizes a secondary objective function based on the desired starting time, Ts.

$$\emptyset(\ t_s) = a|T_s - t_s| + \frac{\gamma}{u_{ij}}\frac{}{t_{d*}} \int_{t_s}^{ts+td*} a *_{ij}\ (t)dt$$

where

$a_{ij}^*(t)$ is minimum cost function to reach node j from node i at time where the selected path may change at each breakpoint,
u_{ij} is a normalizing constant equal to maximum $a_{ij}^*(t)$ over the range (Tmin,Tmax)
Weighting parameter, α, γ,
Jump points t1 < t2 < t3, ..., < tk.

To find the fittest path, perform the following steps such that the primary objective function is minimized:

1. Determine the disjoint intervals in (Tmin, Tmax) where $a_{ij}^*(t)$ is finite.
2. Case 1: If all of the intervals are less than Td, identify the longest interval. In this case td*<Td.
3. Case 2: If some intervals exceed Ts, td* = Td.

In Case 1, evaluate ϕ (ts) for each interval of length td*. Pick the interval with the smallest value of ϕ (ts). For Case 2, apply Theorem 3 of the Phase I Final Report:

Theorem 3: For piecewise-constant $a_{ij}^*(t)$, the optimal start-up time t_s^* that minimizes the secondary objective function is equal to T_s, or $a_{ij}^*(t)$ jumps at t_s^*, or $a_{ij}^*(t)$ jumps at $t_s^*+t_d^*$.

The optimal value is found by performing the following steps:

1. Compute u_{ij} that maximizes $a_{ij}^*(t)$ over (Tmin, Tmax).
2. Compute m_{ij} that minimizes $a_{ij}^*(t)$ over (Tmin, Tmax).
3. Select an interval with jump points {tj} .

4. Define start time evaluation point t_k equal ts where the service starts at T_s and terminates at t_k+T_d.
5. From the jump points, define the origination evaluation points, $\{t_{js}\}$ where the service starts at t_{js} and terminates at $t_{js}+T_d$.
6. From the jump points, define the termination evaluation points, $\{t_{jd}\}$ where the service starts at $t_{jd} - T_d$ and terminates at t_{jd}.
7. Order the solution points defined in steps 3, 4, and 5 to determine the starting and termination points (t_{sk}, t_{dk}) based on the metric: $z_k = | T_s - t_{sk} |$
8. Set ϕmin=+∞;
9. Set k = 0.

Compute loop:

Compute $\phi(t_{s,k},t_{d,k})$ using $a_{ij}^*(t)$ and z_k at jump points

Search Criteria

If $\phi(t_{s,k}, t_{d,k}) < \phi$min, set ϕmin = $\phi(t_{s,k}, t_{d,k})$ and $(ts^*,td^*)=(t_{s,k}, t_{d,k})$;

Stopping Criteria

Stop if there are no more candidates or if

$$\phi(t_{s,k}, t_{d,k}) = z_k + \frac{\gamma}{u_{ij} t_{d,k}} \int_{t_{s,k}}^{t_{s,k}+t_{d,k}} a_{ij}^*(t)dt \le z_{k+1} + \gamma \frac{m_{ij}}{u_{ij}}$$

Else set k = k +1 and continue.

Repeat for additional intervals, if any.

Figure 3-13 depicts an example showing the solution intervals.

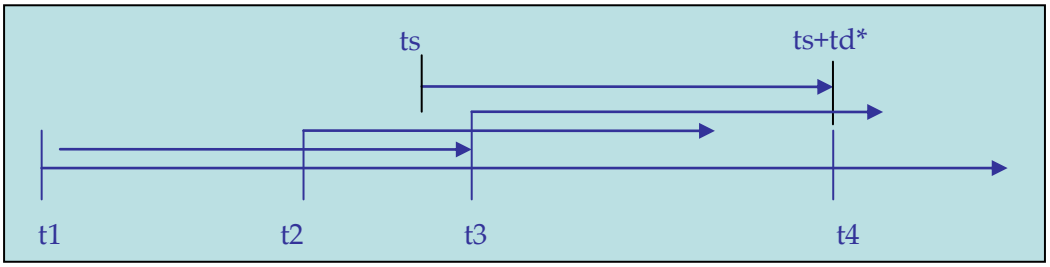

Figure 3-13: Timeliest Fittest Path Example

3.3.1.3 *Two Variable Optimization*

In two-variable optimization, the PCE Optimizer performs optimization on both the starting time and service duration. It uses a parameterized objective function with starting time, duration, and path distance components.

$$\emptyset(t_s, t_d) = \alpha|T_s - t_s| + \beta|T_d - t_d| + \frac{\gamma}{u_{ij}} \frac{}{t_d} \int_{t_s}^{ts+td} a*_{ij}(t)dt$$

Where $0 < t_s$, $0 < t_d < T_d$

Ts, *Td*: desired start-up time and path duration,
$a_{ij}^*(t)$: minimum cost function to reach node *j* from node *i* at time t,
a≥0, *b*>0, *g*≥0: start-up time, duration and path cost weights in objective function
Tmin: earliest starting time
Tmax: latest ending time.

It performs this optimization over the range (Tmin,Tmax) by applying Theorem 4 in the Phase I Final Report:

Theorem 4: For piecewise-constant $a_{ij}^*(t)$, the optimal start-up time t_s^* and duration t_d^* that minimize the objective function and satisfy the following:

1) $t_s^*=T_s$, or $a_{ij}^*(t)$ jumps at t_s^*, or $a_{ij}^*(t)$ jumps at $t_s^*+t_d^*$ and
2) $t_d^*=T_d$, or $a_{ij}^*(t)$ jumps at $t_s^*+t_d^*$.

The optimization is implemented by performing the following steps:

Initialization

1. Determine the candidate solution point intervals based on Theorem 4 : 1.) (Ts,Ts+Td; 2.) start at Ts and terminate at a jump point tk less than Ts+Td; 3.) start at a jump point tk and terminate at Tk+Td or an intermediate jump point; 4.) terminate at either tk+Td or at an intermediate jump point; terminated at a jump point tk and terminate at tk-Td or an intermediate jump point.
2. Determine the normalizing constant, u_{ij} equal to maximum $a_{ij}^*(t)$ over (Tmin, Tmax) where finite.
3. Determine the parameter m_{ij} = minimum $a_{ij}^*(t)$ over (Tmin, Tmax).
4. Order candidate solution points ($t_{s,k}$, $t_{d,k}$) *k*=0,1,2,..., sorted based on increasing order of:
$$z_k = \alpha|T_{sk} - t_{sk}| + \beta|T_{dk} - t_{dk}|$$
with ($t_{s,0}$,$t_{d,0}$)=(*Ts*,*Td*) and z_0=0.
5. Set ϕ min=+∞;
6. Set k = 0.

Compute loop:

Compute $\phi(t_{s,k},t_{d,k})$ using $a_{ij}^*(t)$ and z_k at jump points

Search Criteria

If $\phi(t_{s,k}, t_{d,k}) < \phi$min, set ϕmin = $\phi(t_{s,k}, t_{d,k})$ and (ts^*,td^*)=($t_{s,k}$, $t_{d,k}$);

Stopping Criteria

Stop if there are no more candidates or if

$$\phi(t_{s,k}, t_{d,k}) = z_k + \frac{\gamma}{u_{ij} t_{d,k}} \int_{t_{s,k}}^{t_{s,k}+t_{d,k}} a_{ij}^*(t)dt \le z_{k+1} + \gamma \frac{m_{ij}}{u_{ij}}$$

Else set k = k +1 and continue.

See the Phase I Final Report for an example.

3.3.2 Fixed Path Optimization

3.3.2.1 Zero Variable Optimization

In zero variable optimization, the desired starting time, Ts, and desired duration, Td, must be satisfied exactly. Therefore, the PCE Optimizer checks the generic network has capacity available for the interval (Ts, Ts+Td). It then invokes the PCE Shortest Path module to determine the shortest path between source and destination on this pruned network.

If the PCE Shortest Path is successful and returns a path and cost, this path is optimal. Otherwise, the request is rejected as infeasible.

3.3.2.2 One Variable Optimization

3.3.2.2.1 Timeliest Path

This algorithm is analogous to the switched path algorithm, replacing $a_{ij}^*(t)$ with $c(p_{ij})(t)$ using the secondary objective function:

$$\emptyset(\, t_d) = \beta|T_d - t_d| + \frac{\gamma}{v_{ij}}\frac{}{t_d}\int_{t_s*}^{ts*+td} c(p_{ij})(t)dt$$

vij is a normalizing constant.

3.3.2.2.2 Fittest Path

This algorithm is analogous to the switched path algorithm, replacing $a_{ij}^*(t)$ with $c(p_{ij})(t))$ using the secondary objective function:

$$\emptyset(\, t_s) = a|T_s - t_s| + \frac{\gamma}{v_{ij}}\frac{}{t_{d*}}\int_{t_s}^{ts+td*} c(p_{ij})(t)dt$$

vij is a normalizing constant.

3.3.2.3 Two Variable Optimization

In two-variable optimization for fixed paths, the PCE Optimizer performs optimization on both the starting time and service duration using a slight modification of the switched path algorithm. It uses a similar parameterized objective function with starting time, duration, and path distance components.

$$\emptyset(t_s\,, t_d) = \alpha|T_s - t_s| + \beta|T_d - t_d| + \frac{\gamma}{v_{ij}}\frac{}{t_d}\int_{t_s}^{ts+td} c(p_{ij})(t)dt$$

Ts, *Td* are desired start-up time and path duration,

$c(p_{ij})(t)$ is the time varying cost of the path p_{ij} to reach node *j* from node *i* at time t under the constraint that the same path is used during the service interval,

$\alpha \geq 0$, $\beta > 0$, $\gamma \geq 0$: start-up time, duration and path cost weights in objective function,
v_{ij} is a normalizing constant equal to the maximum cost from i to j over all times in the interval (Tmin, Tmax).

It performs this optimization over the range (Tmin,Tmax).

Initialization

1. Determine the candidate solution point intervals based on Theorem 4 in the final report 1.) (Ts,Ts+Td; 2.) start at Ts and terminate at a jump point tk less than Ts+Td; 3.) start at a jump point tk and terminate at Tk+Td or an intermediate jump point; 4.) terminate at either tk+Td or at an intermediate jump point; terminated at a jump point tk and terminate at tk-Td or an intermediate jump point. [same as for switched path routing]
2. Prune the time dependent generic network to include the bandwidth for each solution point interval.
3. Invoke the shortest path module to determine the cost of the shortest path, c(pij,k)(t), for each interval.
4. Determine the normalizing constant, v_{ij} equal to maximum c(pij,k)(t) over (Tmin, Tmax) where finite.
5. Determine the parameter m_{ij} = minimum c(pij,k)(t) over (Tmin, Tmax).
6. Order candidate solution points $(t_{s,k},t_{d,k})$ k= 0,1,2,..., sorted based on increasing order of:
$$z_k = \alpha|T_{sk} - t_{sk}| + \beta|T_{dk} - t_{dk}|$$
with $(t_{s,0},t_{d,0})=(T_s,T_d)$ and $z_0=0$
7. Set ϕ min=+∞;
8. Set k = 0.

Compute loop:

Compute $\phi(t_{s,k},t_{d,k})$ using c(pij,k)(t) and z_k at jump points

$$\phi(t_{s,k}, t_{d,k} p_{ij,k}) = z_k + \gamma / (v_{ij} t_{d,k}) \int_{t_{s,k}}^{t_{s,k}+t_{d,k}} c(pij,k)(t)dt$$

Search Criteria

If $\phi(t_{s,k},t_{d,k}) < \phi$min, set ϕmin = $\phi(t_{s,k,td,k})$ and $(ts^*,td^*)=(t_{s,k,td,k})$;

Stopping Criteria

Stop if there are no more candidates or if

$$\phi(t_{s,k}, t_{d,k}, p_{ij,k}) = z_k + \frac{\gamma}{v_{ij} t_{d,k}} \int_{t_{s,k}}^{t_{s,k}+t_{d,k}} c(p_{ij,k})(t)dt \leq z_{k+1} + \gamma \frac{m_{ij}}{v_{ij}}$$

Else set k = k +1 and continue.

See the Phase I Final Report for an example.

3.4 Path Selection

Since the shortest path calculation may be performed hundreds of times in order to generate a single optimal-schedule, it is essential that this calculation be done efficiently. The aforementioned Phase I Report described a set of computational techniques for computing shortest paths efficiently. However, for fixed path routing, it was recommended that the Dijkstra algorithm be used. Since Dijkstra was selected for fixed path routing, it was decided to use it for switched path routing as well in this project and defer optimization of the shortest path calculation to later work.

4 PROTOCOLS

4.1 Introduction

The purpose of this section is to specify the extensions to the IETF GMPLS and PCE protocol suites to support scheduled services. Its scope includes enhancements to RSVP-TE, OSPF-TE, and PCE networking protocols as well as the additional objects that will be required in the SNMP Management Information Base (MIB). It is applicable to both Fixed Path (FP) and Switched Path (SP) routing as introduced in the System Architecture section. Also, the enhanced protocols support both scheduled LSPs and on-demand LSPs in the same network using either shared or partitioned capacity as described in the aforementioned architecture section. The algorithmic details required to support these capabilities are described in the Algorithms section above.

The remainder of this section presents the enhancements required by each protocol and discusses special issues. It is organized into the following sections:

- Section 4.2: OSPF Enhancements
- Section 4.3: RSVP-TE Enhancements
- Section 4.4: PCEP Enhancements
- Section 4.5: SNMP MIB Enhancements
- Section 4.6: Special Issues

4.2 OSPF Enhancements

This section describes the enhancements to OSPF to support scheduled LSPs. There are two popular *link state* routing protocols, the *Open Shortest Path First (OSPF)* protocol, and the *Intermediate System-Intermediate System (IS-IS)* protocol, that could be enhanced to support scheduled LSPs. Since OSPF is more popular in commercial networks as well as the dominant choice in research and education networks, it has been selected for use in this project.

However, GMPLS actively supports both OSPF and IS-IS protocols, meaning that any new GMPLS feature that requires routing extensions ultimately defines extensions for both protocols. This work will lead to the straightforward extension to support IS-IS to support scheduled LSPs.

4.2.1 Overview

In order to compute paths for scheduled (and on-demand) LSPs, the path computational element (resident either in a stand-alone platform as in this project or in the NE) needs to know

the network topology and resource information, referred to as the network *traffic engineering (TE) information*. OSPF advertises information on network links (hence the name *link state*) and nodes through information blocks called *Link State Advertisements (LSAs)*.

Neighbor nodes in OSPF form a *routing adjacency* to exchange their LSAs. Through a process known as *reliable flooding*, OSPF protocol guarantees that each node participating in the protocol receives a new or updated LSA at least once. It is this last property of OSPF that makes it an ideal vehicle for distributing *any* piece of information among network nodes, including information that have no meaning, or are *opaque* to the OSPF protocol itself. GMPLS makes use of the OSPF *opaque LSA* option [7.] by inserting traffic engineering information such as link available bandwidth (at each of the eight priority levels that GMPLS supports) and switching capabilities into opaque LSAs, in the form of several concatenated constructs known as *type-length-values* or *TLVs*.

The PCE may obtain the necessary traffic information to compute paths using OSPF in a passive mode, i.e., it executes the protocol with a selected NE, but the PCE does not have any TE links of its own. In general, the PCE may specify the end-to-end path, in full detail, or in an incomplete form consisting of *loose hops*, leaving the *path expansion* to one or more nodes downstream along the path. However, in this single domain application, the PCE will provide the path in full detail in this project.

4.2.2 Scope

For scalability reasons, OSPF has a limited notion of hierarchy as it divides the network (more precisely, the *Autonomous System or AS*, which is the collection of all routers running the same instance of OSPF) into smaller *areas*. By limiting the amount of information exchanged between different OSPF areas, larger networks, possibly with up to tens of thousands of nodes can be supported. Three types of opaque LSAs have been defined, with each type having a different *flooding scope*: Opaque LSAs of type 9 have a *link-local* flooding scope, meaning that they are not flooded beyond the local link or subnetwork [10.]; opaque LSAs of type 10 have an *area-local* flooding scope and are flooded throughout an OSPF area; finally, opaque LSAs of type 11 have the largest flooding scope and are flooded throughout the entire autonomous system [7.].

The algorithms and enhanced protocols developed in this project apply to any type of the aforementioned LSAs within a single domain. It is intended to enhance the algorithms developed in this project to support multiple domains. Rather than require OSPF to distribute TE information between domains, multiple PCEs will be used with each PCE supporting a particular domain. The PCE working group has already developed the OSPF and IS-IS enhancements to support multiple PCEs operating over different domains as discussed in Section 4.4 on the PCE.

One limitation on the use of OSPF is that OSPF provides aggregate traffic loading for each TE link on an aggregate basis, i.e., total number of bits available. It does not provide information on individual LSPs, such as wavelength utilization for all optical networks. For all optical networks enforcing the wavelength continuity constraint, wavelength availability information provides for a more efficient path computation. Without providing the wavelength information, more path computations retries will be required causing excessive signaling crankback.

If such wavelength availability information is to be used for path computation in all-optical networks, the information must be provided by some means over and above the standard

protocols. In this project, the PCE obtains this information using SNMP traps forwarded by the NMS. Refer to sections 4.4 (PCE) and 4.5 (SNMP) for more detail.

4.2.3 Object Description

In accordance with the GMPLS existing mechanism to advertising traffic engineering information for each link, we propose to distribute scheduling-related information through opaque LSAs of type 10, i.e., opaque LSAs with area-local flooding scope. Similar to other traffic engineering information, scheduling information are advertised through a *sub-TLV* within the top-level *link TLV*. Figure 4-1 shows an opaque LSA of type 10 with a top-level link TLV. GMPLS limits the number of link TLVs in each LSA to one, to facilitate quick updates for fine changes in traffic engineering information. A link TLV carries several sub-TLVs carrying various pieces of traffic engineering information defined by RFC 3630 [8.] and RFC 4203 [10.] as well as a new *Timed Interface Switching Capability Descriptor*.

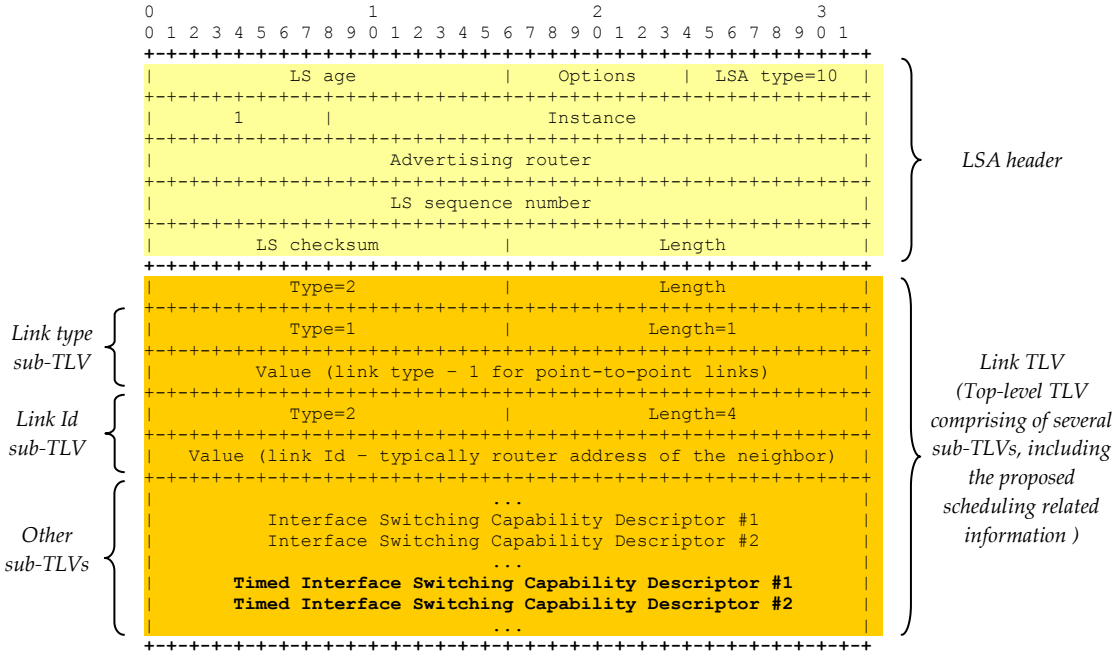

Figure 4-1: Opaque LSA of type 10 with traffic engineering information.

At the time of this writing, a total of 18 sub-TLVs have been defined, all listed in Table 4-1. An up-to-date list can always be found at the IANA registry website [11.]).

Table 4-1: Types for sub-TLVs in a TE Link TLV.

Sub-TLV Type	Sub-TLV Name	Reference
0	Reserved	RFC 3630
1	Link type (1 octet)	RFC 3630
2	Link ID (4 octets)	RFC 3630
3	Local interface IP address (4 octets)	RFC 3630
4	Remote interface IP address (4 octets)	RFC 3630
5	Traffic engineering metric (4 octets)	RFC 3630
6	Maximum bandwidth (4 octets)	RFC 3630
7	Maximum reservable bandwidth (4 octets)	RFC 3630
8	Unreserved bandwidth (32 octets)	RFC 3630
9	Administrative group (4 octets)	RFC 3630

10	Unassigned (Standards Action Required)	-
11	Link Local/Remote Identifiers (8 octets)	RFC 4203
12-13	Unassigned (Standards Action Required)	-
14	Link Protection Type (4 octets)	RFC 4203
15	Interface Switching Capability Descriptor (variable)	RFC 4203
16	Shared Risk Link Group (variable)	RFC 4203
17	Bandwidth Constraints	RFC 4124
18	Neighbor ID	RFC 5329
19	Local Interface IPv6 Address	RFC 5329
20	Remote Interface IPv6 Address	RFC 5329
21-32767	Unassigned (Standards Action Required)	-
32768-32777	Experimental use	RFC 3630
32778-65535	Not Assignable	RF C3630

In particular, the *interface switching capability descriptor (ISCD)* (sub-TLV type 15), shown in Figure 4-2 is the sub-TLV that describes the link available bandwidth, and is the basis for a new sub-TLV that we will define to describe the available bandwidth over time.

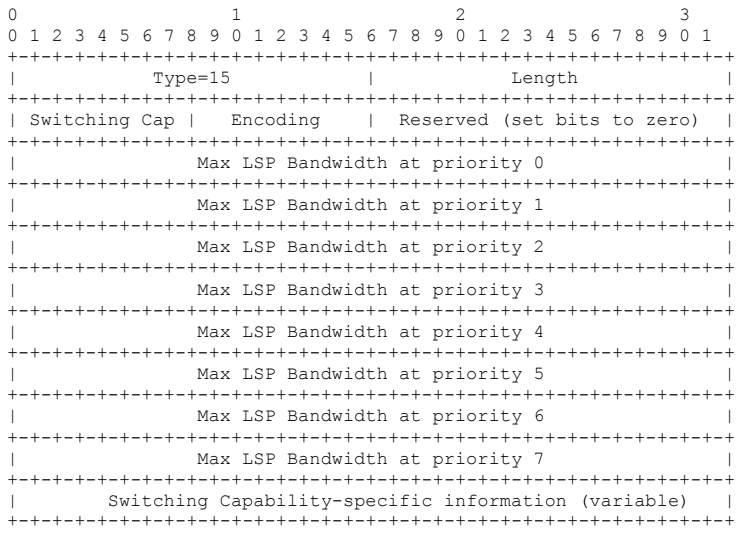

Figure 4-2: Interface switching capability descriptor (ISCD).

The ISCD value fields are defined as follows (see RFC 4203 [10.] for details):

Switching Cap: Defined in RFC 3471 [18], Section 3.1.1, listed here for convenience:

1	Packet-Switch Capable-1 (PSC-1)
2	Packet-Switch Capable-2 (PSC-2)
3	Packet-Switch Capable-3 (PSC-3)
4	Packet-Switch Capable-4 (PSC-4)
51	Layer-2 Switch Capable (L2SC)
100	Time-Division-Multiplex Capable (TDM)
150	Lambda-Switch Capable (LSC)
200	Fiber-Switch Capable (FSC)

Encoding: Defined in RFC 3471 [18] Section 3.1.1, and RFC 4328 [21], Section 3.1.1, listed here for convenience:

1	Packet
2	Ethernet

3	ANSI/ETSI PDH
4	Reserved
5	SDH ITU-T G.707 / SONET ANSI T1.105
6	Reserved
7	Digital Wrapper
8	Lambda (photonic)
9	Fiber
10	Reserved
11	FiberChannel
12	G.709 ODU*k* (Digital Path)
13	G.709 Optical Channel

Max LSP Bandwidth at priority *i*: Maximum LSP bandwidth in *bytes* per second at priority level *i* (*i*=0,1,2,…,7), in the 4-octet IEEE floating point format [8.]; zero is the highest and seven is the lowest priority level.

The last field ("Switching Capability-specific information") is variable-sized and depends on the Switching Cap field. For PSC-1, PSC-2, PSC-3, PSC-4 and TDM switching capabilities, it includes "Minimum LSP Bandwidth" (again in bytes per second, and in the 4-octed IEEE floating point format) to indicate the bandwidth reservation granularity. As of this writing, the field is undefined for other switching technologies, but similar definitions could easily be introduced, should there be a need, particularly for layer-2 switching technologies (L2SC switching capability) such as Ethernet and ATM.

The ISCD sub-TLV can be the basis for the new sub-TLV, *timed interface switching capability descriptor (TISCD)*, to indicate bandwidth availability over time. For all practical purposes, bandwidth reservation and therefore bandwidth availability are piecewise-constant functions of time, which prompts us to express the link available bandwidth by a set of *time-value* pairs $\{(t_1,bw_1),(t_2,bw_2),...\}$. Time intervals are assumed to be closed (inclusive) from the left, and open (exclusive) from the right, i.e., available bandwidth is bw_1 from $t=t_1$ to $t=t_2-1$, and bw_2 from $t=t_2$ to $t=t_3-1$, etc.

Figure 4-3 shows the new proposed sub-TLV. "Switching Cap" and "Encoding" fields have the same definition as the fields with the same name in ISCD. In fact, for each ISCD sub-TLV, a TISCD sub-TLV should be defined if advance reservation is supported for the corresponding resources. Other fields are defined as follows:

Time *i*: Beginning of time interval *i* (*i*=1,…,*n*), using the NTP time format per RFC 1305 [29.] but *without the fractional part*, i.e., number of seconds elapsed since zero hour on January 1, 1900[1].

Max LSP Bandwidth at time *i*: Maximum LSP bandwidth in *bytes* per second for time interval *i* (*i*=1,…,*n*), in the 4-octet IEEE floating point format [8.].

[1] NTP representation of time will see an overflow in 2036, but by following the NTP format, future NTP workarounds can be applied to this representation.

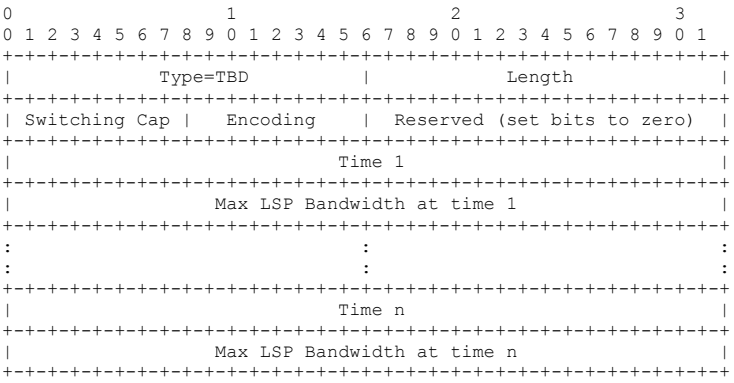

Figure 4-3: Timed interface switching capability descriptor (TISCD).

The option of modifying the timed interface switching capability descriptor to provide availability on a wavelength basis was considered. However, this does not appear practical because the availability TLVs would increase the amount of information advertised by 30-100 times depending on the number of wavelengths per TE link.

4.2.4 Bandwidth Allocation Scenario

The TISCD and ISCD sub-TLVs described above are complementary in respectively representing the bandwidth available to *scheduled connections*, and the bandwidth available to *on-demand connections* (at eight priority levels). This separate representation does not mean that the network bandwidth has to be partitioned for scheduled and on-demand connections. In fact, two reservation models can be defined for co-existence of scheduled and on-demand connections: The *integrated model*, which allocates bandwidth to both scheduled and on-demand connections from the same bandwidth pool, and the *partitioned model*, which allocates bandwidth to scheduled and on-demand connections from separate bandwidth pools.

In integrated reservation model, all link bandwidth is available to both on-demand and scheduled connections. An on-demand connection is assumed to last forever, and any reservation, on-demand or scheduled, affects the available bandwidth in both ISCD and TISCD sub-TLVs. Specifically,

- An on-demand reservation for bandwidth *bw* at priority *p* decreases by *bw* all "Max LSP Bandwidth at priority *i*" fields in ISCD for *i* = *p*,*p*+1,...,7, and decreases by *bw* all "Max LSP Bandwidth at time *i*" fields in the corresponding TISCD[2].
- An advance reservation for bandwidth *bw* decreases by *bw* all "Max LSP Bandwidth at priority *i*" fields in ISCD for *i* = 0,1,...,7, and decreases by *bw* (or modifies) the relevant "Max LSP Bandwidth at time *i*" fields in TISCD.

In partitioned reservation model, on-demand and advance reservations use separate bandwidth pools and there is no coupling between ISCD and TISCD. Specifically,

- An on-demand reservation for bandwidth *bw* at priority *p* decreases by *bw* all "Max LSP Bandwidth at priority *i*" fields in ISCD for *i* = *p*,*p*+1,...,7; no TISCD fields are affected.

[2] Expired time-value pairs are removed from TISCD at every update opportunity; it is not recommended to redistribute an opaque LSA only because one or more time-value pairs in a TISCD have expired (TBD).

- An advance reservation for bandwidth *bw* decreases by *bw* (or modifies) the relevant "Max LSP Bandwidth at time *i*" fields in TISCD; no ISCD fields are affected.

When partitioned bandwidth is used, the NMS must configure the bandwidth allocated to both on demand services and scheduled services. A posible approach is to define separate TE links for each service and define a different administrative color to each TE link.

4.3 Basic Signaling Enhancements

This section describes the GMPLS signaling enhancements to support scheduled services. It presents an overview of the proposed approach, definition of the new protocol objects, signaling scenarios, and the considerations for implementation in all-optical networks.

4.3.1 Overview

Figure 4-4 depicts the key aspects of the RSVP-TE required to support scheduled services using schedule objects that are analogous to standard label objects. As shown in the figure, the PATH message is enhanced with Acceptable Schedule and Suggested Schedule objects while the RESV message is enhanced with a Granted Schedule object.

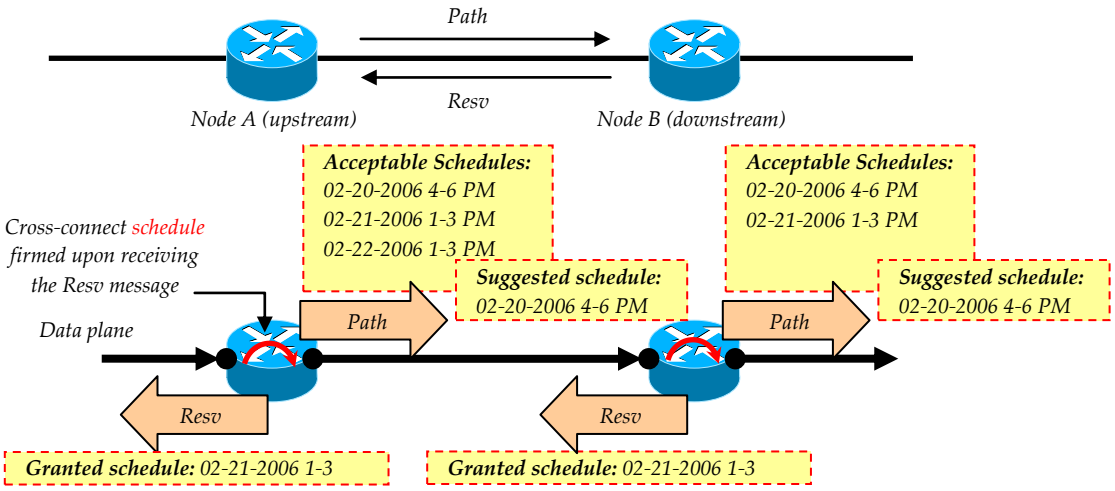

Figure 4-4: Overview of RSVP-TE Signaling for Scheduled Services

In addition, the setting of cross-connects is performed during RESV message processing, as opposed to setting the upstream flow cross-connects during PATH message processing and setting the downstream flow cross-connects during RESV message processing.

The reservation of the cross-connects is different from standard GMPLS to accommodate scheduled bi-directional LSPs. While it results in a delay in setting cross-connects over the standard approach, this delay is inconsequential because data will not begin to flow until sometime in the future. When the cross-connects are physically set during LSP activation, they are set accordance with current GMPLS conventions. This issue is addressed in more detail in Section 4.3.3 (Scenarios) below.

4.3.2 Object Description

This section specifies the new signaling objects for GMPLS scheduled services during LSP set up and activation.

4.3.2.1 LSP Set Up

4.3.2.1.1 SCHEDULE_REQUEST Object

The SCHEDULE_REQUEST object, sent in the Path message, indicates the intention of setting up a scheduled LSP as opposed to the traditional on-demand LSP. As discussed later, this object facilitates interoperability between control plane instances that support scheduling and those that do not support scheduling. The SCHEDULE_REQUEST object has a similar role to the existing GENERALIZED_LABEL_REQUEST object, and has to be processed in conjunction with that object. Specifically, data plane requirements (LSP switching type, LSP encoding type, and generalized payload identifier (G-PID)) are still extracted from the GENERALIZED_LABEL_REQUEST object, and the SCHEDULE_REQUEST object is added to indicate the scheduled nature of the LSP.

The SCHEDULE_REQUEST object is shown in Figure 4-5. The object fields are defined as follows:

Length: Total object length in bytes, a multiple of 4 and currently 4 bytes.

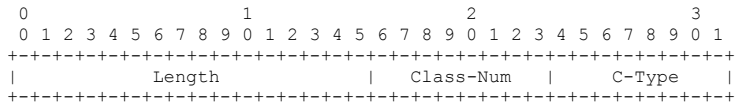

Figure 4-5: SCHEDULE_REQUEST object.

The proposed Class-Num and C-Type for the object are 240 and 1 respectively.

As described in Section 5, the PCEP also requires a SCHEULE_REQUEST object. However, the PCEP request requires the additional parameters: desired start time, desired duration, earliest start time, latest start time, minimum duration, and desired duration. The expanded format that includes these parameters is specified in Section 5.

4.3.2.1.2 SCHEDULE_SET Object

The SCHEDELE_SET object, sent in the Path message, describes a set of time schedules acceptable or unacceptable to the upstream node, similar to the LABEL_SET object that describes a set of labels acceptable or unacceptable to the upstream node. Representing a set of schedules is not as straightforward as labels however. Labels come from a *one-dimensional* label space and can be conveniently aggregated by a *range*. On the other hand, time schedules come from a *two-dimensional* space with dimensions of start-up time t_s and path duration t_d, which suggests representing a continuous set of schedules by a *region* in the t_s-t_d plane. Therefore, both explicit lists of schedules (similar to labels) and schedule regions (new for schedules) are supported in the SCHEDULE_SET object. In this object, a *schedule region* is defined as a *closed (inclusive) polygon in the t_s-t_d plane*, and is specified by the list of the polygon vertices traversed in the clockwise direction, starting from an arbitrary vertex.

The SCHEDULE_SET object is shown in Figure 4-6 where the object fields are defined as follows:

Length: Total object length in bytes, a multiple of 4 and at least 8 bytes.

Action: Defines how the schedules in the object should be interpreted. It assumes one of the following four values:

0 Inclusive list: schedules define a *list* of acceptable schedules.

1 Exclusive list: schedules define a *list* of unacceptable schedules.

2 Inclusive region: schedules define the vertices of a closed polygon in a plane with dimensions of start-up time and path duration, traversed in clockwise direction; all schedules inside or on the border of the polygon are acceptable.

3 Exclusive region: schedules define the vertices of a closed polygon in a plane with dimensions of start-up time and path duration, traversed in clockwise direction; all schedules inside or on the border of the polygon are unacceptable.

Start-up time *i* (*i*=1,…,*n*): Start-up time for *i*-th schedule, in the standard NTP time format [29.], but with the fractional part removed, i.e., the number of seconds elapsed since zero hour on January 1, 1900.

Path duration *i* (*i*=1,…,*n*): Path duration for *i*-th schedule, in seconds.

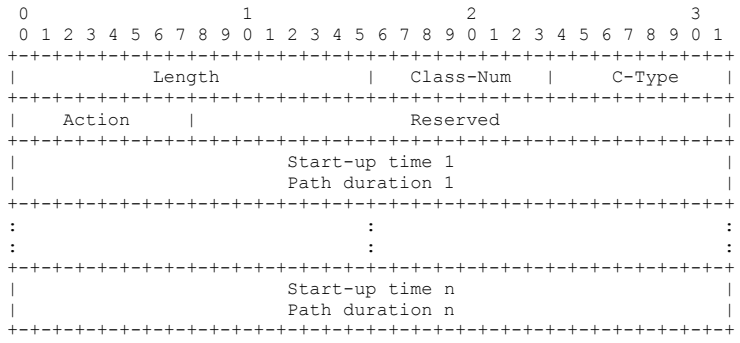

Figure 4-6: SCHEDULE_SET object.

The proposed Class-Num and C-Type for the object are 241 and 1 respectively.

The following example illustrates the representation of a schedule region.

Example 4-1: Assume that the cost of a given path from ingress node *i* to egress node *j* is as shown in Figure 4-7(a). Further assume that the scheduling requirements are $T_{s,min} \le t_s \le T_{s,max}$ and $T_{d,min} \le t_d \le T_{d,max}$. Combining the requirements, we see that the earliest start-up time for the path is $T_{s,min}$, and the latest start-up time is $T_{s,max} - T_{d,min}$. The allowable schedule region is the convex polygon shown in Figure 4-7(b), together with one of the four possible representations of the polygon.

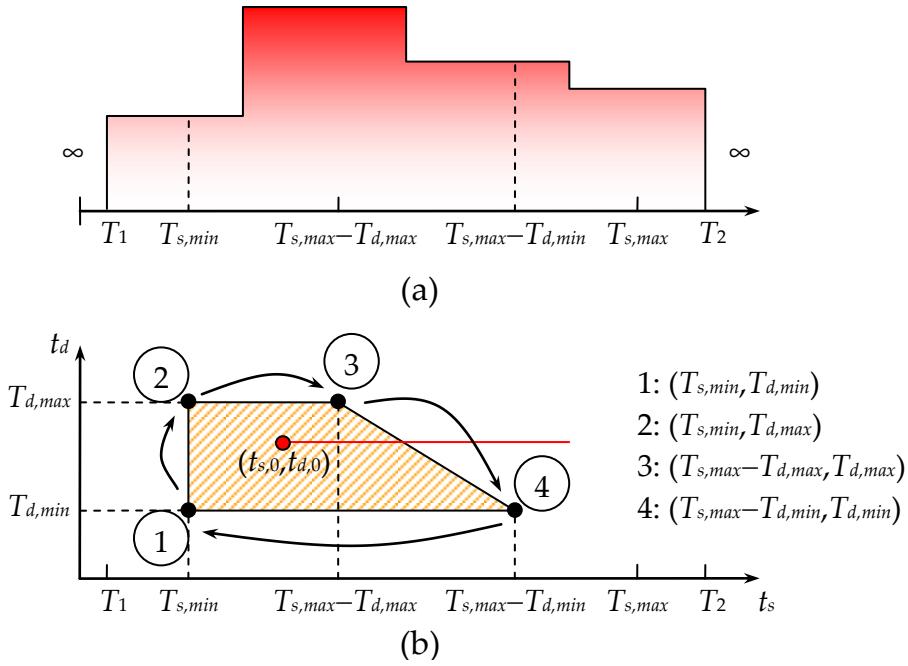

Figure 4-7: Example of path availability and the corresponding schedule region.

Determining if a given schedule ($t_{s,0}$, $t_{d,0}$) is acceptable to the upstream node or not is equivalent to finding if the point ($t_{s,0}$,$t_{d,0}$) is inside or on the polygon representing the acceptable schedule[3]. This is a classical problem in computer graphics, with several algorithmic solutions in the literature. A simple algorithm is as follows: Consider the horizontal ray emanating from ($t_{s,0}$,$t_{d,0}$) to the right (see Figure 4-7 (b)). If the number of intersections with the polygon boundaries is even (including zero), the point is outside the polygon. Alternatively, if the number of intersections is odd, the point is inside the polygon.

Since the algorithms developed for this project as described in Section 3 do not generate such regions, the capability to support regions is viewed as a future capability.

4.3.2.1.3 SUGGESTED_SCHEDULE Object

The SUGGESTED_SCHEDULE object, sent in the Path message, declares a time schedule suggested by the upstream node, similar to the SUGGESTED_LABEL object that declares a label preferred by the upstream node. The object is shown in Figure 4-8 and its fields are defined as follows:

Length:	Total object length in bytes, which is 12.
Start-up time:	Start-up time for the suggested schedule, in the standard NTP time format [29.], but with the fractional part removed, i.e., the number of seconds elapsed since zero hour on January 1, 1900.
Path duration:	Path duration for the suggested schedule, in seconds.

[3] The acceptable schedule may consist of multiple non-overlapping polygons in the t_s-t_d plane; in this case, a separate test is done for each polygon.

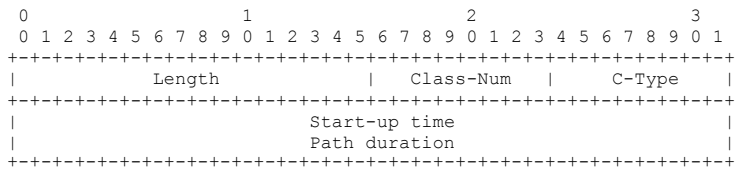

Figure 4-8: SUGGESTED_SCHEDULE object.

The proposed Class-Num and C-Type for the object are 242 and 1 respectively.

4.3.2.1.4 ACCEPTABLE_SCHEDULE_SET Object

The ACCEPTABLE_SCHEDELE_SET object, sent in the PathErr message, describes a set of time schedules acceptable or unacceptable to the downstream node, analogous to the ACCEPTABLE_LABEL_SET object that describes a set of labels acceptable or unacceptable to the downstream node. As described earlier, SCHEDULE_SET and ACCEPTABLE_SCHEDULE_SET objects provide a schedule negotiation capability between two neighbor nodes.

The object definition is identical to the SCHEDULE_SET object, except for its proposed Class-Num, whose value is 243.

4.3.2.1.5 RSVP_SCHEDULE Object (Granted Schedule)

The RSVP_SCHEDULE object, sent in the Resv message, includes the schedule confirmed and recorded by the downstream node, similar to the RSVP_LABEL object, which communicates the label allocated by the downstream node.

The object definition is identical to the SUGGESTED_SCHEDULE object, except for its proposed Class-Num, whose value is 244.

4.3.2.1.6 Mapping of Schedules and Labels

As described above, the new schedule objects introduced are analogous to the current label objects. Table 4-2 summarizes this comparison. The major difference between label objects and schedule objects is that different labels may be used in the upstream and downstream directions, but the same schedule must be used in both directions, i.e., the upstream and downstream data flows must occur at the same time. This difference impacts the approach used for setting cross-connects as described in Section 4.3.3.1.

Table 4-2: Label and Schedule Object Comparison

Label Objects	**Schedule Objects**	**Relevant Message**
UPSTREAM_LABEL	None	PATH
LABEL_SET	SCHEDULE_SET	PATH
SUGGESTED_LABEL	SUGGESTED_SCHEDULE	PATH
ACCEPTABLE_LABEL	ACCEPTABLE_SCHEDULE	PATHERR, RESVERR
GENERALIZED_LABEL	RSVP_SCHEDULE (Granted)	RESV

Both label and schedule objects have corresponding suggested and acceptable objects that may contain multiple labels or schedule components. So a convention must be established for the specifying the corresponding validity of labels and schedules. For example, when the SUGGESTED_LABEL and SUGGESTED_SCHEDULE contain an inclusive list, are all labels valid for all schedules? Table 4-3 summarizes the cases that must be considered and the proposed convention.

Table 4-3: Inclusive-Exclusive Conventions

Action	Label Actions	Action	Schedule Actions	Convention
0	Inclusive List	0	Inclusive List	All labels applicable to all schedules
		1	Exclusive List	Exclusive List Schedule – Not Used
		2	Inclusive Region	All labels applicable to region
		3	Exclusive Region	Exclusive Region Schedule – Not Used
1	Exclusive List	0	Inclusive List	All labels excluded to all schedules
		1	Exclusive List	Exclusive List Schedule – Not Used
		2	Inclusive Region	All labels excluded to region
		3	Exclusive Region	Exclusive Region Schedule – Not Used
2	Inclusive Range	0	Inclusive List	All labels applicable to all schedules
		1	Exclusive List	Exclusive List Schedule – Not Used
		2	Inclusive Region	All labels applicable to region
		3	Exclusive Region	Exclusive Region Schedule – Not Used
3	Exclusive Range	0	Inclusive List	All labels excluded to all schedules
		1	Exclusive List	Exclusive List Schedule – Not Used
		2	Inclusive Region	All labels excluded to region
		3	Exclusive Region	Exclusive Region Schedule – Not Used

A special case to consider is highlighted in the table where both an inclusive list of labels and schedules is provided. Suppose both lists contain N elements. Rather than use the all labels apply to all schedules convention as shown in the table, it may be preferred that the i^{th} label be feasible only for the i^{th} schedule. This option is introduced by setting the schedule action field to a value of 5.

4.3.2.2 LSP Activation

4.3.2.2.1 Overview

A protocol design choice in distributed path scheduling is the *mechanism of path activation*, or how to coordinate the timing (if any need at all) of the label allocations at different nodes along the path. There are clearly two approaches to activating a scheduled path:

1) Uncoordinated activation: Since each node knows the resource to be allocated and the associated schedule, each node autonomously allocates the label (i.e., makes cross-connections or creates forwarding entries in the data plane) at start-up time, or *activation time*, and autonomously de-allocates the label (i.e., removes data plane cross-connections or forwarding entries) once the path has been active for the decided duration, or at *de-activation time*. Thus, the end-to-end connection is automatically available during the scheduled time, with no extra signaling effort.
2) Coordinated activation: Although each node knows the resource to be allocated and the associated schedule, nodes wait for an explicit activation command, initiated by the ingress node, and allocate labels in a specific and deterministic order. Thus, the end-to-end connection becomes available after an extra signaling step.

The first approach is clearly simpler, as it relies on each node to activate its scheduled reservation, which the node is already aware of. This is technically possible, except for the fact that in the absence of perfect time synchronization between nodes (which is always the case), the order of activation depends on the local clock at each node; i.e., the cross-connections or forwarding entries in the data plane are made in an unpredictable order. This is an undesirable outcome for at least two reasons:

1) The ingress node needs to know when the data path is ready, in its entirety, to start data transmission.
2) Other nodes need to assume a certain order of activation in the data plane to be able to draw a meaningful conclusion with respect to real data plane failures; for example, in normal RSVP-TE, every node sending a Resv message upstream can safely assume that all its downstream nodes have completed their label allocation, and the data path to the egress node must be active in the absence of network failures.

For these reasons, and generally to bring determinism into the set up procedure, we take the second approach, and *serialize* the activation procedure by initiating an explicit activation request at the ingress node. The idea is to forward the request all the way to the egress node, and sequentially activate the scheduled allocation at the nodes, starting with the egress node and moving in the upstream direction. This is done through two new RSVP-TE objects, as described in the following subsections, but there are no existing RSVP-TE objects that correspond to these new objects.

4.3.2.2.2 REQUEST_ACTIVATION Object

The REQUEST_ACTIVATION object, sent in the Path message, and initiated by the ingress node, is an explicit indication to each node along the path to initiate the state machine that ultimately activates the scheduled reservation. The object is shown in Figure 4-9 and its fields are defined as follows:

Length:	Total object length in bytes, which is currently 8.
MLT:	Maximum lead time - the maximum waiting time (in seconds) that the node can hold an activation request before activating its scheduled reservation; otherwise the node must send a ResvErr to the node requesting the activation.

Reserved: Reserved for future extensions; should be set to zero when sending the object and must be ignored when receiving the object.

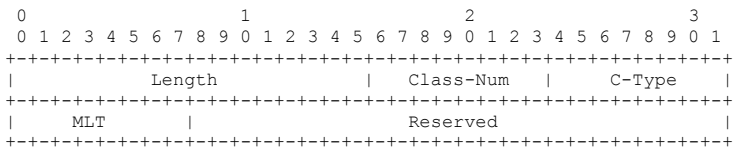

Figure 4-9: REQUEST_ACTIVATION object.

For an example of how Maximum lead time is used see the example scenario below. The proposed Class-Num and C-Type for the object are 245 and 1 respectively.

4.3.2.2.3 ACTIVATE_SCHEDULE Object

The ACTIVATE_SCHEDULE object, sent in the Resv message, indicates that the reservation schedule in the sending node has been successfully activated. The upstream node receiving this object attempts to activate its own schedule. If activation is successful, the node sends a ACTIVATE_SCHEDULE object in the Resv message to its own upstream; otherwise, the node sends a ResvErr downstream, indication that activation was unsuccessful. The downstream node receiving the ResvErr, may choose to send another Resv (and possibly repeat it multiple times), or abandon the activation process altogether and remove the path.

The ACTIVATE_SCHEDULE object is shown in Figure 4-10 and its fields are defined as follows:

Length: Total object length in bytes, with a minimum of 4.

Options: Optional fields for future extensions.

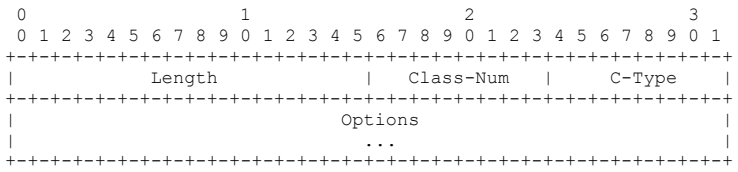

Figure 4-10: ACTIVATE_SCHEDULE object.

The proposed Class-Num and C-Type for the object are 246 and 1 respectively.

If the LSP activation fails, the node nearest to both the failure point and the ingress node sends a GMPLS notify message to the ingress node. The ingress node will make the decision whether to release the LSP or retry.

Note that this is a "slowest clock approach" where the activation will be delayed until the time in the node with the slowest clock reaches the activation time. It is expected that this delay will be small compared to the scheduling granularity, i.e., the time between the end of an existing LSP and the beginning of a new one. Therefore, the delay will not introduce any resource conflicts between old and new LSPs.

The most probable cause for activation failure is discrepancy between the neighbor nodes local clocks. To illustrate, consider the scenario where a connection is scheduled to start at t_s, but because of differences in local clocks a transit node receives an ACTIVATE_SCHEDULE object ahead of the schedule (according to the node local clock), at $t_s - \Delta$. To simplify the arbitration

process for multiple LSPs using the same resource at different scheduled intervals, we require the node *not* to activate its schedule before t_s (according to its own local clock); instead the node does one of the following:

- If the lead time Δ is sufficiently small (specifically, smaller than or equal to the maximum lead time defined in the REQUEST_ACTIVATION object), the node simply waits for Δ time units and activates the schedule at t_s, according to its own local clock. The node then sends an ACTIVATE_SCHEDULE object to its upstream to continue the activation process.
- If the lead time Δ is large (specifically, greater than the maximum lead time defined in the REQUEST_ACTIVATION object), the node sends a ResvErr message downstream with an IPv4 or IPv6 IF_ID ERROR_SPEC object with Error Code 2 (Policy control failure) [14.] and a new Error Value with the meaning "Activation Error/Lead Time Too Large" (for the most recent list of Error Value sub-codes for Error Code 2 see the IANA RSVP registry [20.]).

When an activation failure occurs, the ingress node may request activation for second time (or for a defined number of times) by resending the REQUEST_ACTIVATION object after a defined wait time. The number of activation attempts is a configurable parameter.

4.3.3 Resource Scheduling Scenarios

4.3.3.1 Set Up of Bi-Directional LSPs

RSVP protocol was originally designed to reserve network resources for IP flows. To simplify protocol design, IP flows were considered to be *unidirectional*, which was not limiting at all, as bidirectional communication could be accomplished through separate (and more importantly *independent*) IP flows in each direction of the communication. All later protocol enhancements, developed under the official name of RSVP-TE, stayed faithful to the unidirectional reservation style of RSVP, evident by the fact that MPLS LSPs are unidirectional. Since GMPLS had a major emphasis on transport networks, bidirectional reservation became important for at least two reasons:

- Transport networks such as SONET/SDH and photonic networks are often designed for bidirectional communication, making it difficult to allocate resources in only one direction of communication without wasting the mirror resources in the opposite direction.
- Even if transport networks can efficiently support unidirectional resource allocation, there is a strong interest in bidirectional reservation for easier control and management. For example, it is often desired to have *fate sharing* for the two directions of a bidirectional service, i.e., should the service fail in one direction, it is often desired to terminate the service in both directions, and migrate both direction to a new network path.

GMPLS does not modify the unidirectional nature of a label; however, it adds support for bidirectional LSPs by allowing two labels to be allocated by each node along the bidirectional LSP, the (traditional) downstream label corresponding to data transmission from ingress to egress, as well as an *upstream label*, which defines a network resource in the opposite direction. Observing that in RSVP the downstream label is allocated by the downstream node, designers of the new protocol chose to put the upstream label allocation in the hands of the *upstream* node. Specifically, in bidirectional LSP set up, the upstream node allocates the upstream label prior to

sending the Path message, and the downstream node allocates the downstream label prior to sending the Resv message.

Figure 4-11 presents an example of label negotiation for a bidirectional LSP set up. In this example, the upstream node (node A) unilaterally selects an upstream label (label u_0) and sends it through an UPSTREAM_LABEL object in the Path message. The upstream object is not acceptable to the downstream node (Node B) however, which results in a PathErr message containing an ACCEPTABLE_LABEL_SET object. In response, the upstream node selects a different upstream label (label u_1) and sends a new UPSTREAM_LABEL object in the Path message. The downstream label is allocated by Node B before sending a Resv message upstream.

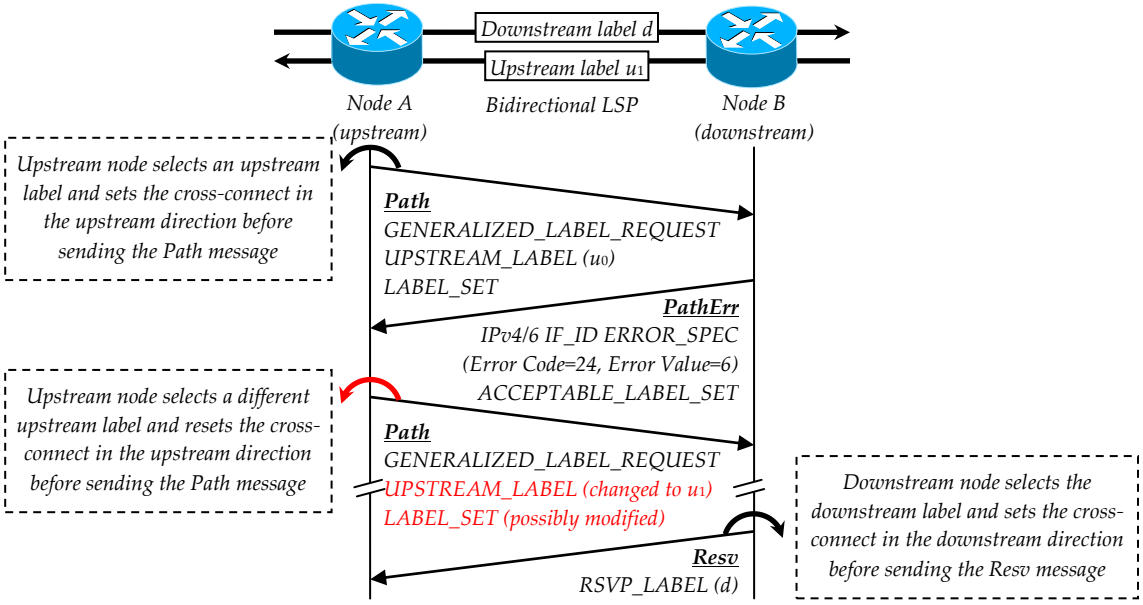

Figure 4-11: Standard Label negotiation and selection for bidirectional LSPs.

In extending the label paradigm to support scheduled LSPs, there are two key issues that must be addressed: label symmetry and label selection. Regarding label symmetry, there is no way to tell if the labels in an ACCEPTABLE_LABEL_SET object in the PathErr message are acceptable for the *upstream* (reverse) direction (which will prompt the upstream node to select a different upstream label), or they are acceptable for the *downstream* (forward) direction (which will prompt the upstream node to modify its LABEL_SET object), or *both*.

Since PathErr is originated by the downstream node to suggest a label for the flow controlled by the downstream but must be selected by the upstream node (downstream direction), it is assumed that it indicates an acceptable downstream label (as done for unidirectional flows). Therefore, it implies no information implied regarding the upstream direction flow. Thus, it does not aid the upstream node in revising its UPSTREAM_LABEL to be used in the next Path message

This ambiguity is best resolved by defining a new UPSTREAM_ACCEPTABLE_LABEL_SET object. Until then, an ACCEPTABLE_LABEL_SET object in a PathErr message should be assumed to *apply to both directions*, i.e., the upstream node should match *both* the selected upstream label *and* the labels in the LABEL_SET object against the received acceptable list.

As to label selection, the current label allocation scheme for bidirectional LSPs is *asymmetric in the way nodes can influence label selection in the upstream and downstream directions*. For label allocation in the downstream direction, all nodes may influence the selected schedule, as the reservation intent is known to all of them in the form of a GENERALIZED_LABEL_REQUEST object in the Path message. For example, if labels have global significance and label continuity is desired (e.g., in photonic networks with transparent wavelength switching, or Ethernet networks with transparent VLAN switching), nodes have a chance to limit the first allocated label in the downstream direction (the label allocated by the egress node) by passing more restrictive LABEL_SET objects downstream. In contrast, label allocation in the upstream direction is driven by the upstream node during the Path message processing, which leaves no room for all involved nodes to influence the label allocation *before any upstream label allocation takes place*. As a result, in the absence of a complete view of available network resources and label switching constraints in the nodes along a bidirectional LSP, several *crankbacks* may be necessary before label allocation in the upstream direction is completed, with each crankback likely to cause routing updates.

Extending the above label allocation scheme to scheduling, there are two ways to schedule bidirectional LSPs:

1) in direct correspondence with RSVP-TE label allocation, upstream scheduling and label negotiation can be done during Path processing, and downstream scheduling and label negotiation can be done during Resv processing, or

2) with a slight departure from RSVP-TE label allocation, both upstream and downstream scheduling and label allocation can be done during Resv processing.

The first approach is a direct extension of the RSVP-TE reservation style; crankback may be reduced or even eliminated when the schedule and corresponding labels are selected prior to signaling, using an up-to-date view of the network resource information. However, in the absence of perfect knowledge of network resources (e.g., when resource and information is stale, or at domain border nodes), crankback is inevitable.

The second approach, depicted in Figure 4-12, defers the scheduling and label allocation for the upstream direction to the time when Resv message is processed, which means that all nodes along the path have had a chance to participate in scheduling by sending a SCHEDULE_SET object downstream.

The second approach has been selected for the following reasons:

1) for a bidirectional LSP, labels for upstream and downstream direction could be different, *but schedules are the same*[4]. Thus, a good protocol design should prevent mechanisms that could potentially allow different schedules to be selected for upstream and downstream directions,

2) by selecting the schedule and upstream labels during Resv processing, all nodes are given a chance to influence the choice of schedule, as well as both downstream and upstream labels; and

3) this is not a fundamental departure from the RSVP-TE reservation style, *because it applies to a scheduled connection, which is going to carry data in the future*. In other words, the activation of a

[4] Different schedules in each direction can be simply accommodated by two unidirectional LSPs.

scheduled bidirectional LSP can still follow the RSVP-TE style: Scheduled resources in the upstream direction can be activated prior to sending a REQUEST_ACTIVATION object downstream, and scheduled resources in the downstream direction can be activated prior to sending an ACTIVATE_SCHEDULE object upstream.

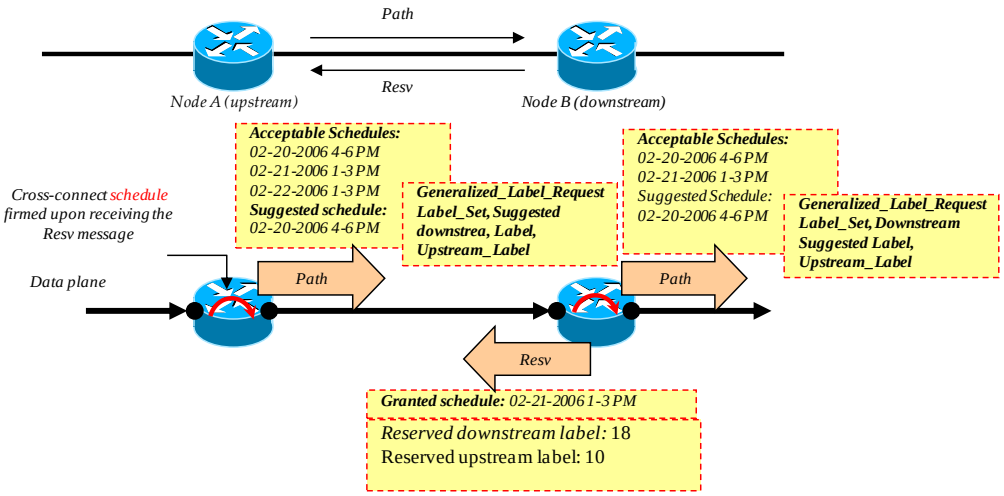

Figure 4-12: Bi-directional LSP Set Up - Reservation Phase

4.3.3.2 *Set Up of Uni-Directional LSPs*

The set up of uni-directional scheduled LSPs will be handled as a special case of a bi-directional scheduled LSP. In this case, the cross-connects will be reserved on the Resv message path (as opposed to the Path message path used for standard GMPLS)

4.3.3.3 *LSP Activation*

Figure 4-13 depicts an LSP activation scenario beginning at time *t*=10 (local node A time) when node A initiates the activation process by sending a REQUEST_ACTIVATION object in the Path message. The object is quickly passed through all nodes, making them aware of the maximum lead time for the scheduled connection. At *t*=10 local node D time, node D activates its own scheduled reservation and sends an ACTIVATE_SCHEDULE object upstream in a Resv message. Upon receiving the Resv message, node C immediately activates its scheduled reservation, as it has received the activation request after the scheduled activation time (according to its own local clock), and sends an ACTIVATE_SCHEDULE object upstream to node B. For node B however, the activation request arrives too much ahead of time; specifically, since the request is made more than two seconds before the scheduled activation time (i.e., lead time of more than two seconds), node B rejects the request by immediately sending a ResvErr back to Node C. Node C chooses to wait for a defined period (two seconds in this example), and reattempts the activation by sending another ACTIVATE_SCHEDULE object in a Resv message. Node B accepts the request this time, activates its own scheduled reservation, and passes an

ACTIVATE_SCHEDULE object to its upstream, which is the ingress node. The ingress node activates its own internal scheduled reservation (if any), and begins data transmission[5].

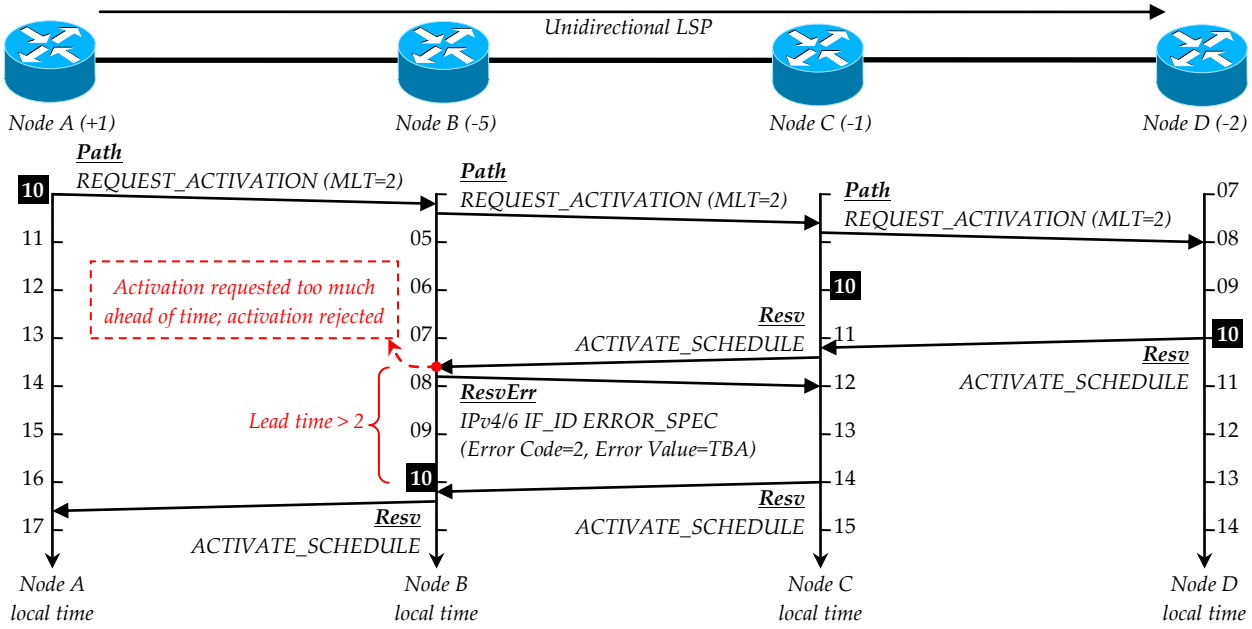

Figure 4-13: Example of schedule activation with imperfect time synchronization.

If the activation fails, the node closest to the failure and the ingress node sends a notify message to the ingress node. Then the ingress node will decide whether to retry the activation or release the LSP. For example, the activation may have failed because some other LSP may not have released the resources required to activate the LSP.

The logic for the activation of bi-directional LSPs is identical.

4.3.4 All Optical Network Considerations

As addressed in Section 4.3.2, there must be a method to determine the labels that may be used for each schedule when multiple labels and multiple schedules are provided. In all optical networks, this association is an especially complex issue because the selection of a label, i.e., wavelength, has downstream significance, not just local significance.

Because of the complexity of associating wavelengths and schedules, it may be desirable in an all-otpical networks to provide only a SUGGESTED_SCHEDULE with one or more labels (wavelengths) valid for the schedule. The SCHEDULE_SET object would not be used.

[5] A protocol design alternative is to hold the early activation requests, *in the forward phase*, while passing the REQUEST_ACTIVATION object in the Path message. Our approach causes fewer activation reattempts and has faster overall activation time, because it pipelines the activation requests and gives more catch up time to nodes whose local times are behind.

4.4 PCE Protocol Enhancements

This section describes the enhancements to the PCEP to support scheduled services in the "stateful" mode. In addition it describes the wavelength availability information that the NMS may provide for the all optical switching technology case.

4.4.1 Statelful Operation

In the "stateful" operation mode, the PCE maintains the status of all LSPs rather than just the bandwidth availability on each TE link. In order to track the LSP status, the PCE receives status updates of all LSP requests from the NMS, i.e., whether the LSP was successfully set up, failed, or released, and updates its Traffic Engineering Database (TED). With this information, the PCE is able to compute the wavelength (or wavelength set) as well as compute path and schedule set in response to requests of the PCC in the NMS.

When the PCE is operating in the "stateful" mode and receives wavelength availability from the NMS, it does not require OSPF to provide bandwidth availability using either the existing interface switching capability descriptor or the new timed interface switching capability descriptor. However, the PCE does require OSPF to provide LSA configuration information including Router IDs for both endpoints, TE link interface IDs or IP addresses, interface switching descriptors, SRLGs, administrative colors, and TE link engineering metric (link cost).

If the "stateless" mode is implemented for all-optical networks, LSP set up failure is more likely to occur because the PCE has not determined if any wavelength is available to support the schedule. In this mode, the PCE computes the path and schedule using bandwidth availability information provided by the timed interface switching descriptor independent of wavelength while the network elements negotiate the wavelengths. Using the computed path and schedule, the ingress node determines the available labels for the first link on the path and forwards them downstream using RSVP-TE signaling. At each transit node, the network element narrows down the set of acceptable labels as signaling proceeds downstream.

If a label can be selected at the egress node, then the forward path has been successful and signaling on the reverse path may begin. If the acceptable label set becomes empty, the network element adjacent to the link where the ultimate failure occurred generates a GMPLS notify message. This message provides the details on what failed (node ID, TE interface) so failure cause can be provided to NMS and PCE for input to path computation retry. This is done using an exclude object.

The PCE will then retry, e.g., by excluding the TE link where the failure occurred. However, this approach does not guarantee that a path-schedule will be found if one exists and is not recommended.

While the "stateful" mode is being used in this project, the "stateless" mode is generally suitable for other switching technologies, e.g., packet, Ethernet, that may have a very large number of LSPs.

4.4.2 Protocol Concepts

The PCE Protocol (PCEP) is a request-response protocol that enables the PCC (resident in the NMS or NE) to provide path-schedule requests to the PCE and receive responses. In the schedule request, it provides the standard PCEP parameters for generating LSP paths (endpoint IP addresses, data rates, etc.) and the new scheduled service parameters:

Desired start time and desired duration,
Earliest and latest start time,
Minimum service duration,
Suggested wavelength (label) for all-optical networks with fixed transponders,
Label Set for all-optical networks with tunable transponders.

In response, the PCE provides the path, wavelength in the case of all optical switching, LSP start time, and LSP duration. For switched paths, the PCE provides the schedule parameters for each of the constituent LSPs.

As an option, the PCC may provide the endpoint IP address as well as the endpoint interface. This will allow the PCE to track the availability of the endpoint interfaces connecting external devices or systems.

In the general case, the PCE may generate one or more schedules and associated paths. Therefore, the PCEP response message also requires the introduction of the SCHEDULE_SET and SUGGESTED_SCHEDULE objects as defined for RSVP-TE. In addition for all optical networks, the PCE may also provide a LABEL_SET and/or SUGGESTED_LABEL as defined for RSVP_TE. For bidirectional LSPs, it also assumed that these objects apply in both directions.

When a scheduled LSP set up failure occurs, the NMS may request the PCE to compute a new path excluding the TE link where the set up failure occurred. To exclude this TE link, the NMS uses the PCE exclude object (XRO) as specified in [28].

4.4.3 Object Description

This section describes the new objects that are introduced into the PCEP.

4.4.3.1 Request Objects

The PCEP request is modified by the addition of a SCHEDULE_REQUEST and a DESIRED_SCHEDULE object. The SCHEDULE_REQUEST object that is identical to SCHEDULE_REQUEST object in the Path message used in signaling while the DESIRED_SCHEDULE includes five additional parameters:

Desired start time,
Desired duration,
Earliest start time,
Latest start time,
Minimum acceptable duration.

Figure 4-14 depicts the format of the DESIRED_SCHEDULE object. If any of the time parameters are not provided, zeroes should be included for the corresponding field. The standard NTP time representation is used for the start up time fields and the duration time fields are specified in seconds.

Length: Total object length in bytes, a multiple of 4 bytes and currently 24 bytes.

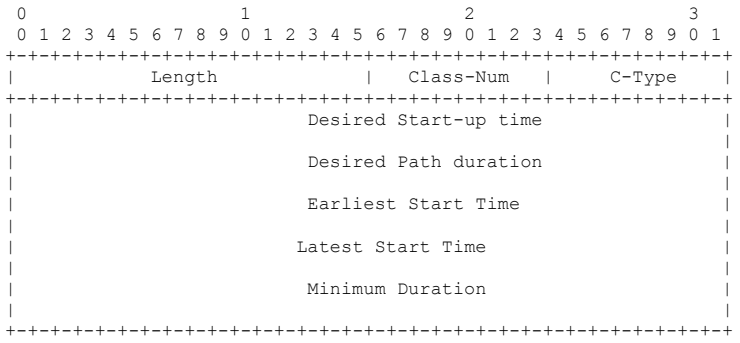

Figure 4-14: DESIRED SCHEDULE object

The proposed Class-Num and C-Type for the object are 247 and 1 respectively.

For all-optical networks, the schedule request may also include a SUGGESTED_LABEL, e.g., when fixed wavelength transponders are used. This object is the same as the RSVP_TE SUGGESTED_LABEL object. If tunable transponders are used, the schedule request may include the set of wavelengths used for scheduled services. This is provided by the RSVP_TE LABEL_SET object.

When the PCE is computing a schedule-path in response to a crankback request, the PCC will also include the PCE Exclude Route Object (XRO) as defined in [28]. The link to be excluded is selected using heuristic logic, e.g., the link where the reservation failure occurred is excluded.

4.4.3.2 Response Objects

The PCEP response message requires a SCHEDULE_RESPONSE object the introduction of the SUGGESTED_SCHEDULE and SCHEDULE_SET objects defined above for the Path message. There are no changes.

The SCHEDULE_RESPONSE object has the same format as the SCHEDULE_RESPONSE object with a new C-TYPE. If SP routing is being, the SCHEDULE_RESPONSE must also include the number of path segments as shown in Figure 4-15. The object fields are defined as follows:

Length: Total object length in bytes, a multiple of 4 and currently 4 bytes.

NS: Number of path segments

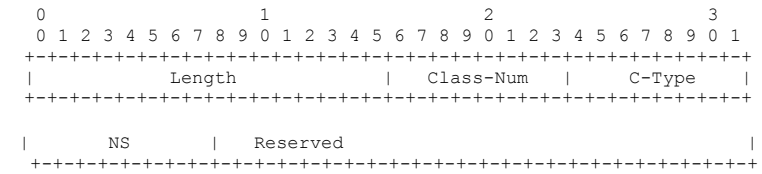

Figure 4-15: SCHEDULE_RESPONSE Object

The proposed Class-Num and C-Type for the object are 248 and 1 respectively.

For bidirectional paths, the SUGGESTED_SCHEDULE and SCHEDULE_SET objects apply in both directions. For all-optical networks, the PCE will also generate the wavelength. This field is returned to PCC using RSVP_TE LABEL_SET.

4.4.3.3 Event Message Objects

The Event Message is a new PCEP message that enables the PCC to provide the state information to the PCE. It includes the RSVP_TE SESSION_OBJECT and a new DISPOSITION object shown in Figure 4-16.

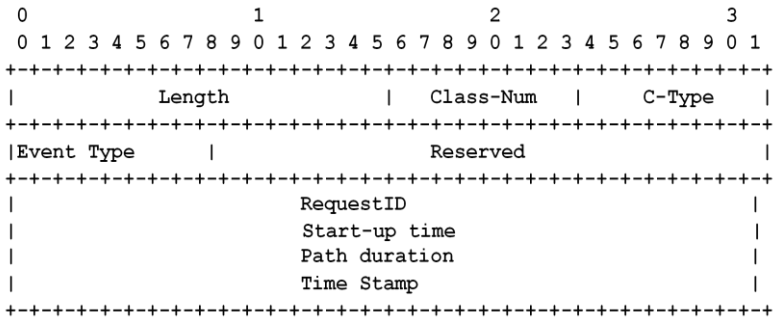

Figure 4-16: Disposition Object Format

The object fields are defined as follows:

Length:	Total object length in bytes, a multiple of 4 and currently 4 bytes.
Event Type:	1 LSP Set Up Success
	2 LSP Set Up Fail
	3 LSP Activation Success
	4 LSP Activation Fail
	5 LSP Release Success
	6 LSP Release Fail
	7 LSP Path Switchover (future if needed for SP routing)
	8 LSP Path Switchover (future if needed for SP routing)
RequestID	PCEP RequestID
Start Up Time	Actual start time in NTP format
Path Duration	Actual duration in NTP format
Time Stamp	Time of Event in NTP format

The proposed Class-Num and C-Type for the object are 249 and 1 respectively.

For SP routing, the Start Up Time and Duration correspond to the relevant path segment, not the entire LSP.

When nodes or components fail or are set administratively out of service, e.g. interfaces, the NMS will notify the PCE via the PCC using PCEP. When the components are repaired or returned to service, the NMS will notify the PCE. The objects used for notification are TBD.

If the PCE fails catastrophically, it would retrieve the LSP state from the NMS. Details are will b specified in Phase III.

4.4.4 PCE Scenario

This section describes a scenario showing the interaction of the PCC, PCE, NMS, and NE in the set up of a scheduled LSP. As depicted in Figure 4-17, the PCE obtains network information from the NEs in the form of OSPF LSAs.

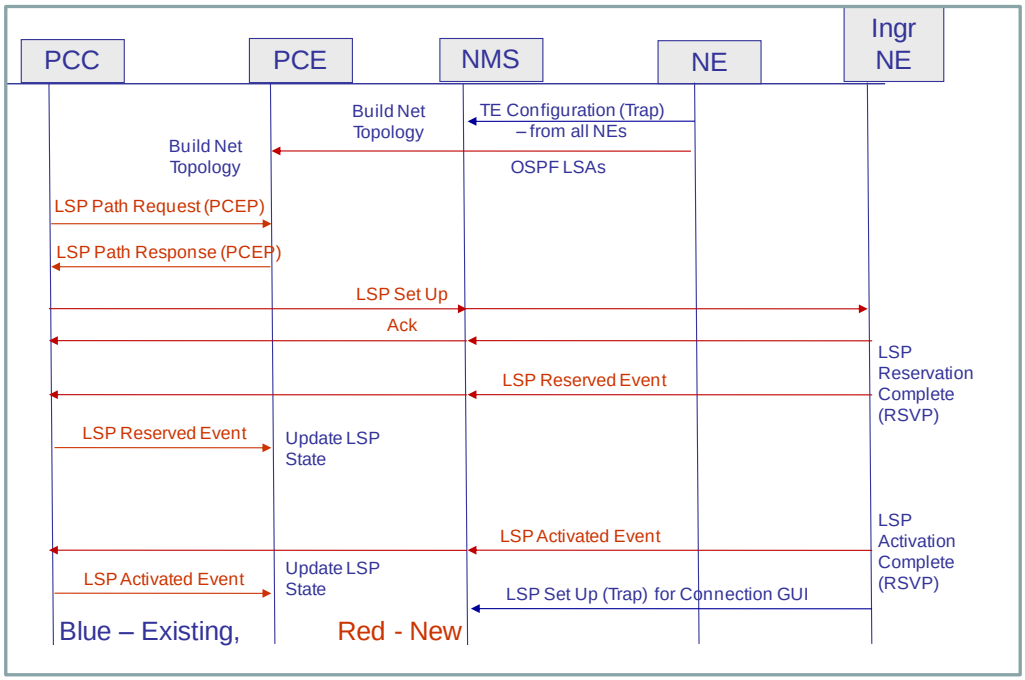

Figure 4-17: Scheduled LSP Path Generation, Set Up, and Activation

As user requests for service are received, the PCC requests the PCE to compute a schedule and path. Then the PCC forwards the results to the NMS to establish the path by reserving resources for use at the desired start time. When the start time occurs, the NE activates the resources without any assistance from the NMS.

In order to track the LSP state, the NE informs the NMS when the resources are reserved and activated for the LSP. The NMS informs the PCE of these state changes via PCC using PCEP.

4.5 SNMP Enhancements

This section describes the modification of the LOPSYS MIB to support scheduled services for using all-optical networks. In the future, the modifications to the standard MIBs will be specified. There are no changes to SNMP.

4.5.1 Overview

The SNMP is a request-response protocol enabling the NMS to manage the network elements. It also provides a notify message allowing the network elements to informs the NMS of events as they occur.

4.5.2 Object Description

This section describes the specific objects required to support scheduled services.

4.5.2.1 LSP Objects

The following new attributes are associated with each LSP

Start time
Duration

For SP routing, this information must be provided for each segment.

Also, the LSP state must be enhanced to support pending and activate states. The Switchover from one path segment to another path segment in SP routing may also require another state.

4.5.2.2 TE Link Objects

Each TE link requires a timed interface switching capability descriptor that requires the following parameters:

Time of bandwidth availability change,
Bandwidth value.

Also for the partitioned bandwidth option, the administrative color (though not a new parameter) may now be used to indicate TE links supporting scheduled or on demand services. This parameter may be set or retrieved.

4.5.3 Scenario – Notify Messages

The operation of SNMP is unchanged. However, it will require new values in the SNMP notify messages.

4.6 Special Issues

This section addresses interoperability and SP re-routing issues that were not addressed in the previous sections.

4.6.1 Interoperability with Standard GMPLS Systems

4.6.1.1 Interoperability Approach

It it is essential to make sure that a combination of nodes supporting and *not supporting* scheduled LSPs consistently and reasonably well together. Specifically, we have established the following two requirements in designing the signaling extensions,

- Any two end nodes, supporting or not supporting the scheduling extensions, *must* be able to set up and tear down on-demand LSPs between them, through nodes that may or may not support the scheduling extensions.
- Any two end nodes supporting the scheduling extensions *must* be able to set up and tear down scheduled LSPs between them, through nodes that may or may not support the scheduling extensions.

The first requirement is trivially satisfied: In the absence of a SCHEDULE_REQUEST object in the Path message, reservation is considered on-demand, and label allocation is accomplished by following the normal protocol procedures. The second requirement is also satisfied, once we observe that nodes not supporting scheduling treat an advance reservation as a normal (on-demand) reservation, and immediately allocate labels that will remain allocated until the scheduled part of the path is activated and released.

4.6.1.2 *Interoperability Scenario*

To illustrate the second requirement, consider the scenario shown in Figure 4-18, where ingress node A is setting up a scheduled unidirectional connection to egress node D. Both ingress and egress nodes support the scheduling extensions, but transit nodes B and C do not.

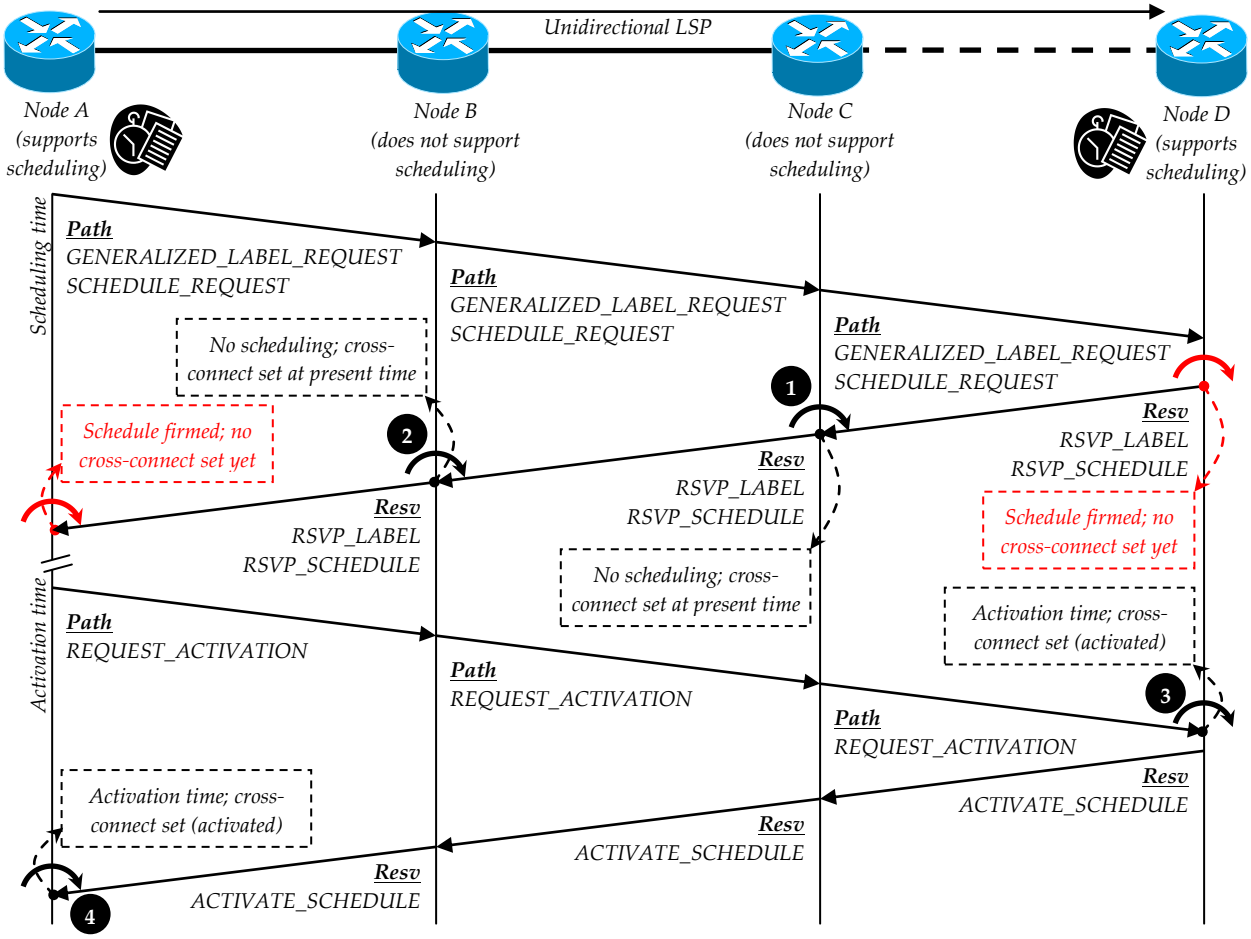

Figure 4-18: Interoperability with Nodes Not Supporting Scheduled LSPs

As a result, the transit nodes interpret the reservation request as immediate reservation, and initialize the label allocation state machine. They also do not recognize the SCHEDULE_REQUEST and other scheduling-related objects in the Path message, but forward them transparently downstream. Upon receiving the Path message, node D recognizes the scheduling objects, selects a label on link C-D and the reservation schedule associated with the label, and communicates them upstream through RSVP_LABEL and RSVP_SCHEDULE objects. Again oblivious to the passed schedule, nodes C and B allocate labels on links B-C and A-B *upon receiving a Resv message* (events 1 and 2 in Figure 4-18), and forward the RSVP_SCHEDULE object transparently upstream. Upon receiving a Resv message, node A recognizes the RSVP_SCHEDULE object, reserves its own internal resources (if any), and completes the set up procedure. At this point in time, *the A-B-C part of the path is activated*, although it is not going to carry data until the scheduled start-up (activation) time. When the activation time arrives, node A sends a REQUEST_ACTIVATION object downstream, which is passed transparently to node

D. In response, node D activates the label reservation on link C-D (event 3), and sends an ACTIVATE_SCHEDULE object upstream, which is transparently passed to the ingress. At this point in time, node A activates its own internal resources (if any), and data transmission can be started (event 4).

With regards to routing, the ISCD and TISCW must also be updated. By convention, RFC 3630 [8], the NEs advertise the bandwidth available on the TE link from itself to its neighbor. For this example of a uni-directional LSP from A to D, the TE links A->B, B->C, and C->D must be updated by A, B, and C, respectively.

Therefore, NE B and C will update the available bandwidth field for traffic-engineering links B->C and C->D at the time of initial scheduling; these two links do not have a TISCD field. Since NE A is schedule aware, it will update the TISCD field for link A->B is updated at the time of scheduling and the ISCD field for link A->B the time of activation.

4.6.2 Re-routing of Switched Paths

When the duration of a SP routing consists of multiple segments, the transfer of service from the current LSP segment to the successor LSP segment uses a "make before break" switchover. This switchover uses the standard GMPLS control plane "make before break" protocols [15]. While the details of the data plane operation are necessarily proprietary, there is nothing new here. However, the path segments will have different LSP IDs.

The following sections describe the operation of the control plane and data plane to support "make before break." Note for this case of "make before break", it is necessary to minimize the service disruptions because no failure has occurred – as opposed to make before break in response to failures where a service disruption has occurred.

The following sections describe the general control plane actions, the data plane actions for an all-optical network, and resulting conclusions.

4.6.2.1 Control Plane

The control plane capabilities to support "make before break" involve only the RSVP-TE signaling protocol. According to the GMPLS protocol standard [15], make before break uses the following procedures.

Session attribute defining the tunnel endpoints (in both Path and Resv messages) is the same for both the current and successor LSPs.

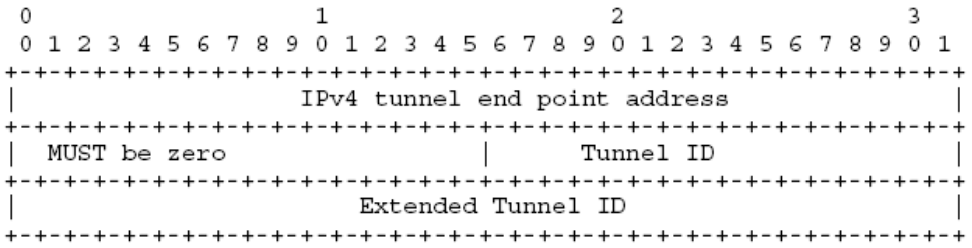

Figure 4-19: Session Attribute

Sender template in the Path message is used to distinguish the current and successor LSPs by using different LSP Ids. FilterSpec distinguishes the LSPs in the Resv message in the same way.

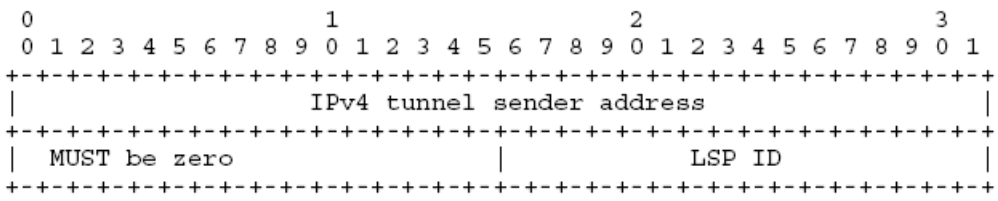

Figure 4-20: Sender Template

Flag in the session attribute in the Path message is set (0x04) in the flags field to allow sharing of resources using the shared explicit (SE) style. It indicates that the ingress node may choose to re-route this LSP without tearing it down. The egress node should set the option vector in the style object transmitted in the Resv message to use the SE mode.

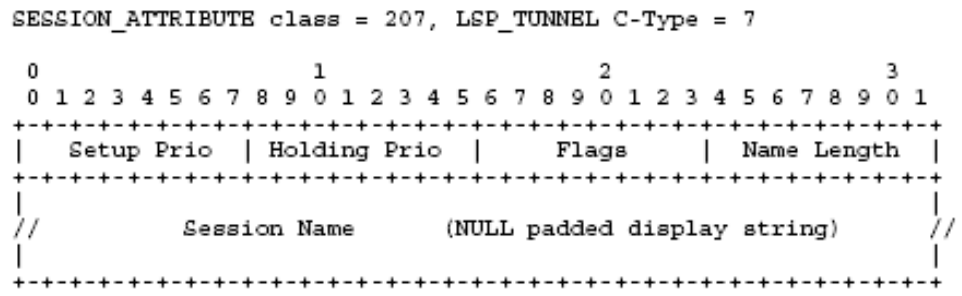

Figure 4-21: Session Attribute Flag

To set the option vector SE, bits and 20 and 21 are set to 10b and bits 22-24 are to 010b.

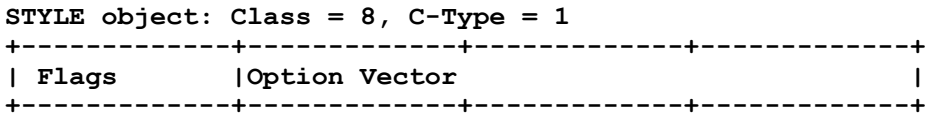

Figure 4-22: Style Object

In summary, the use of these objects allows the control plane to have the current and successor LSPs share common resources where appropriate.

4.6.2.2 Data Plane

The following example illustrates the operation of the data plane to support make before break in an LN2000 network for a uni-directional LSP. Figure 4-23 depicts the flows of the current LSP (LSP1:A-B-D-E-G) and the successor LSP (LSP2: A-B-D-G).

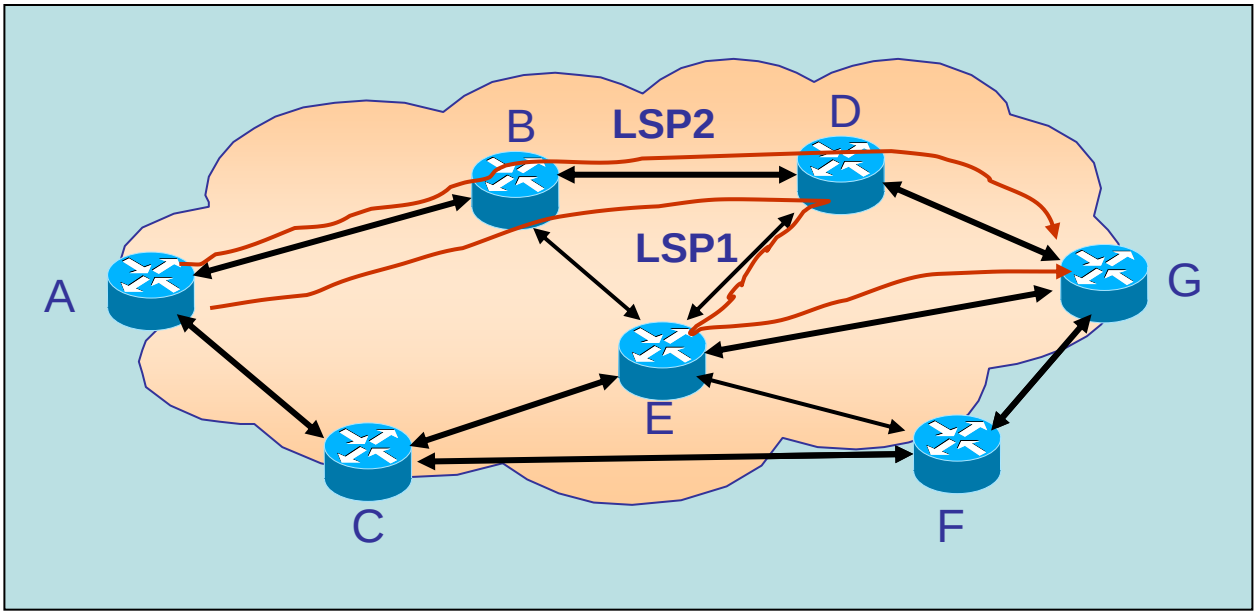

Figure 4-23: Make Before Break Example

The signaling and associated data plane operations for set up of LSP2 and release of LSP1 are depicted in Figure 4-24. As depicted in the figure, common resources are shared on the A-B-D path; fabric bridging is required at D and resetting the OWI receive selectors upon detecting a good signal is done at G.

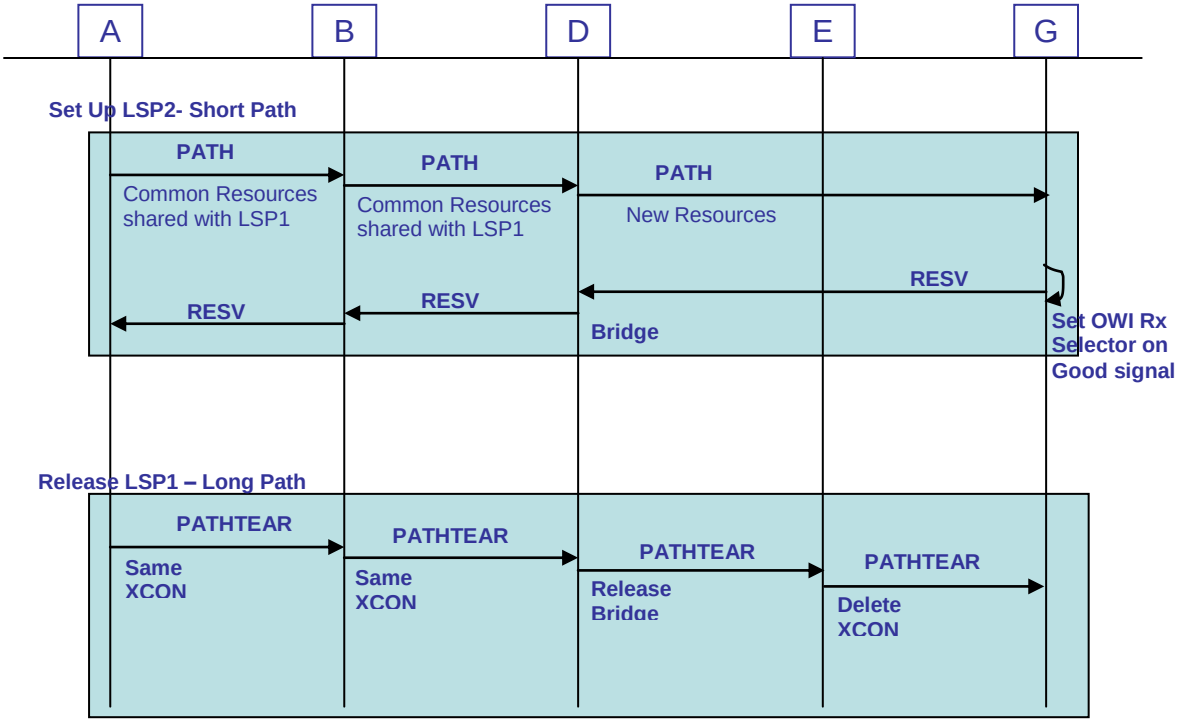

Figure 4-24: Make Before Break Operations

In order to bridge the signal at D, the redundant LN2000 switch fabrics are used as depicted in Figure 4-25. As shown the side0 of the WOSF switches the signal to WMX8 while side 1

switches the signal to WMX7. This allows BOSF0 to forward the signal to TPM2 and BOSF1 to forward the signal to TPM3. When LSP1 is released, the cross-connects supporting it may be reset to provide redundant operation for LSP2.

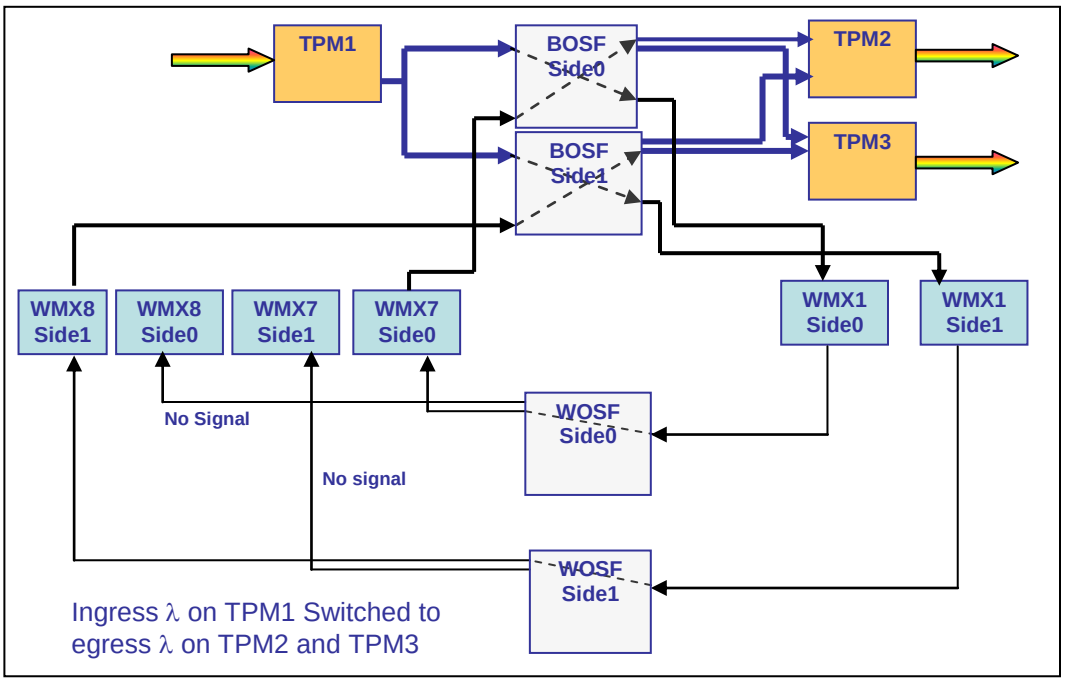

Figure 4-25: Signal Bridging at D

After the signal is bridged and the second path set up, node G will be receiving two good signals. It will reset its OWI selectors to receive the second path.

4.6.2.3 Bi-directional Make Before Break

The support of bi-directional LSPs in this example will require additional actions in the reverse direction (G->A). In the reverse direction G-A, the bridging is performed at the OWI on node G as depicted in Figure 4-26. As depicted in the figure, side0 switches the signal to WMX7 and side1 switches the signal to WMX8 allowing BOSF0 to forward the signal to TPM2 and BOSF1 to forward the signal to TPM3.

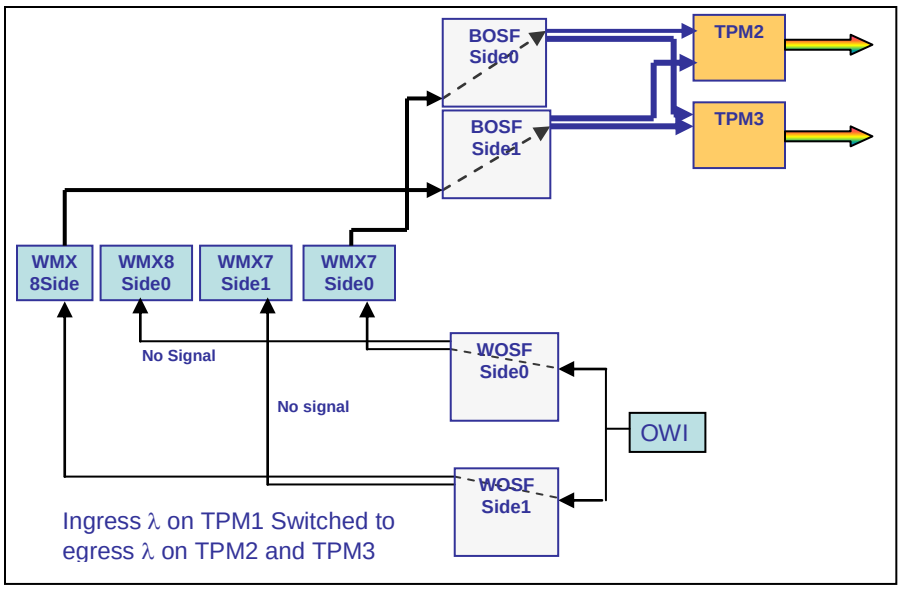

Figure 4-26: OWI Bridging at G for Reverse Flow G->A

At node D, merging of the two signals is required. As shown in Figure 4-27, in order to perform the merging, node D uses side1 of the fabric to detect the LSP2 signal. When a good signal is received on side1, the TPM selectors may be set to side1.

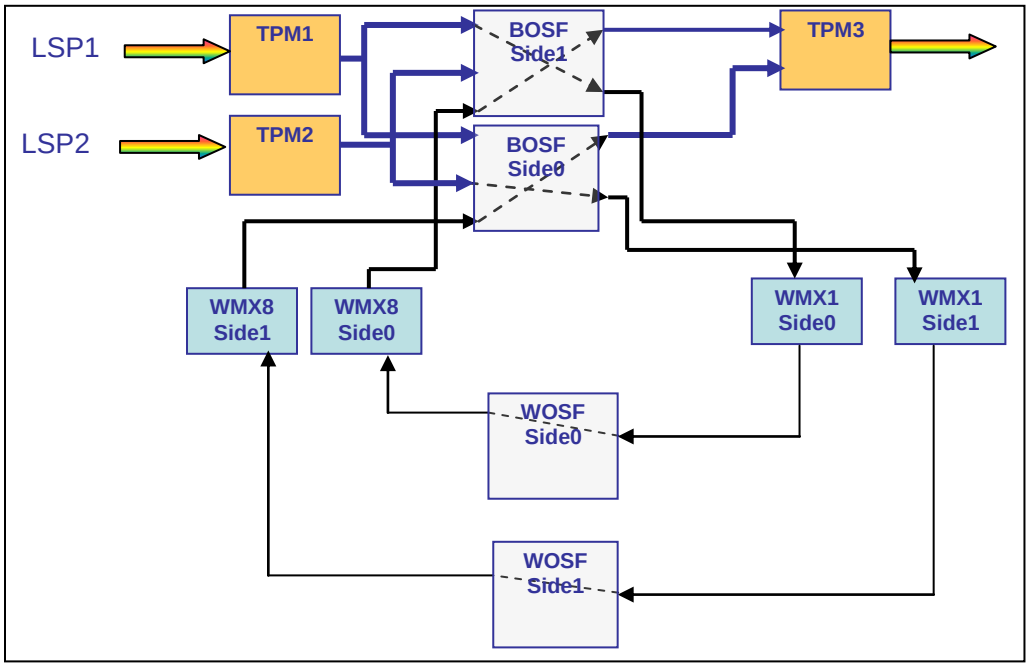

Figure 4-27: Merging Flows at D for Reverse Flow G->A

4.6.2.4 SP Routing Conclusions

The example described above illustrates that "make-before-break" signaling is feasible in all-optical networks to support SP routing. However, it would use fabric bridging in some switching architectures such as the LN2000 to maintain the old and new LSPs concurrently. This

would not be an attractive approach because the node would sacrifice its switching redundancy during its "make-before-break" operation. Furthermore, as described in Section 2.4.3, circular dependencies may occur in some instances of SP routing preventing a smooth switchover.

Therefore, SP routing is better suited for packet technologies where TE links may be overloaded than optical technologies where physical resources must be allocated during the switchover.

5 PERFORMANCE ANALYSIS

5.1 Introduction

This section presents the results of the empirical performance studies carried out using a selected set of the GMPLS scheduling algorithms. These algorithms perform the route generation, wavelength assignment, and scheduling of time dependent label switched paths (LSPs). The details of the System Architecture, Algorithms, and Protocols modeled in this section are presented in Sections 2, 3, and 4 above, respectively.

The purpose of this performance study is to assess whether these algorithms are suitable for use in commercial GMPLS scheduling software products. Its scope encompasses packet and lambda switching technologies, Fixed and Switched Path routing techniques, on demand and threshold routing approaches, as well as sensitivity to algorithm and network parameters.

These algorithms have been implemented in a standalone Path Computational Engine (PCE) platform using a Personal Computer (PC) running the Linux operating system. The hardware elements included a 1.5 GHz processor, 1024 KB cache, and 512 MB RAM. With Linux implementation, the Phase II PCE algorithms will be easily ported to a variety of operational platforms for Phase III, e.g., Network Management System (NMS), Network Element (NE), or stand-alone platform. Thus, our software approach provides a strong foundation for product development.

As depicted in Figure 5-1, this platform supports the PCE as well as a Traffic Engineering Database (TED). The major enhancement in this TED over typical GMPLS TEDs is that the scheduling enhancements have been implemented. These enhancements include new attributes (actual and desired starting times, actual and desired durations, TE utilization by time) as well as the support for both Fixed Path and Switched Path LSPs.

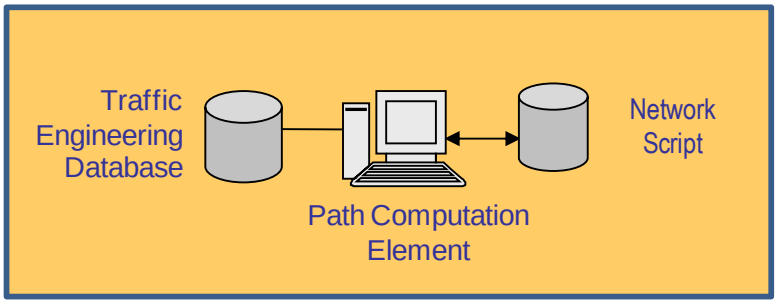

Figure 5-1: Algorithm Evaluation System

Since the PCE is operating in a stand-alone mode independent of any NMS or NEs, it is driven by a Network Script. This script provides the network topology (nodes, TE links) and

simulates the submission of user requests with the associated LSP attributes (source-destination pair, desired start time, desired duration, data rate or wavelength).

The following sections first present a description of the test case networks in Section 5.2 and then present a set of five test case sensitivity studies in Sections 5.3 to 5.7. Then Section 5.8 summarizes the conclusions of this performance study.

5.2 Test Case Networks

This section describes the test networks as well as the associated traffic and network/algorithm parameters.

5.2.1 Network Description

The networks and parameters used in the performance studies are described in the following sections including the switching technology, network topology, and bandwidth.

5.2.1.1 Switching Technologies

While the algorithms developed during this project may be applied to any GMPLS switching technology, they have been implemented and tested for lambda and packet switching technology because they are two of the most common GMPLS switching technologies.

5.2.1.2 Topology

The performance studies have been carried out using three networks , T1, T2, and T3 having 7, 15, and 32 nodes with the corresponding number of Traffic Engineering (TE) links shown in Table 5-1. The rationale is to begin testing with a small network and then evolve the modeling to represent a typical metro size network having 15-32 nodes.

Table 5-1: Topology Data

Topology	T1	T2	T3
Nodes	7	15	32
TE Links	10	26	56

5.2.1.3 TE Bandwidth

The algorithms support both bandwidth allocation techniques:

- partitioned – separate bandwidth allocation for On Demand LSPs and Scheduled LSPs,
- shared – both types of LSPs utilize the same bandwidth.

While the algorithms support both options, only the partitioned technique was modeled because the thrust of this effort is on the performance of the algorithms for scheduled LSPs. The shared bandwidth option can be added at a later time using the same PCE for both on demand LSPs and scheduled LSPs. For on demand LSPs, the desired start time would be set to the current time and the duration would be set arbitrarily long.

The number of wavelengths (lambdas) allocated to scheduling depends on the specific application and customer requirements. Table 5-2 enumerates the parameters used in this study.

For evaluation of lambda switching technology, the capacity of each TE link was assumed to be 2 lambdas as a specific case. It enables a fair comparison evaluation of the comparison of the algorithms because it is desired to determine CPU time per LSP rather than an aggregate value to schedule a batch of LSPs on all wavelengths. It was assumed that any wavelength could be used, i.e., tunable transponders, and all wavelengths were checked to determine the optimal solution. If more wavelengths were used, the CPU time per LSP would increase, e.g. approximately x4 if 8 wavelengths were available instead of 2 wavelengths.

For packet switching, the capacity of each TE link was assumed to be 10 Mbps. For most cases, the algorithms allow the use of the complete 10 Mbps except if a threshold is imposed. If a threshold is imposed (0.5), the threshold bandwidth (0.5x10 Mbps = 5 Mbps) may not be exceeded. However, the threshold may be enforced in either a soft (after the LSP is allocated) or hard (before the LSP is allocated) manner.

Table 5-2: TE Link Capacity Parameters

Switching Technology	Capacity	Shortest Path Re-compute Threshold
Lambda	2 Lambdas/TE Link	Not Applicable
Packet	10 Mbps/TE Link	0.5

5.2.2 Traffic

The traffic was controlled by using arrival rate and service duration parameters over a non-dimensional time space, referred to as grid time units. In this way the results may be applied independent of whether the scheduling time horizon is hours, days, or weeks.

5.2.2.1 LSP Arrival Rates

A set of arrival rates was defined to correspond to a very heavy (Rate1), heavy (Rate2), and medium (Rate3) traffic loadings for both lambda and packet traffic. While these traffic rates exceed that carried in operational networks due to the large number of blocked requests, they are useful for analysis purposes to compare the algorithms under different situations.

Table 5-3 enumerates the traffic parameters initially used in the study. Later when testing the packet threshold algorithm with a threshold of 0.5, a fourth traffic rate was added to accommodate the reduced capacity with the threshold. It provided an increased arrival rate with a lower bandwidth per LSP (refer to Section 5.2.2.3 for LSP bandwidth parameters).

Table 5-3: Traffic Parameters

Rate1	0.5 LSP requests/unit time
Rate2	0.375 LSP requests/unit time
Rate3	0.25 LSP requests/unit time
Rate4	2.0 LSP requests/unit time

In generating traffic between nodes, a uniform traffic pattern was assumed. In this traffic model, the source node is generated by selecting 1 of N nodes with probability 1/N. Then the destination node is generated by selecting 1 of the remaining N-1 nodes with probability 1/(N-1).

Since the traffic patterns are random, three traffic files were generated for each case. Then the algorithms were executed for each traffic file and the results averaged to generated statistics.

5.2.2.2 LSP Service Duration

The LSP service duration was generated as a uniform random variable with:

Minimum Duration: 10 time units,

Maximum Duration: 40 time units.

Use of these parameters results in a mean service duration of 25 ((10+40)/2) time units (used in all test cases).

5.2.2.3 LSP Bandwidth

For packet switching the LSP bandwidth was uniformly generated between 0 and 10 Mbps for most cases. When performing the sensitivity studies on the threshold algorithm, the bandwidth was uniformly generated between 0 and 2 Mbps.

For wavelength switching, it is assumed that an LSP utilizes a complete wavelength so the full bandwidth of the wavelength is available to the LSP.

5.2.3 Parameters

The controllable used in the study are the routing weights in the objective function and the start time window.

5.2.3.1 Routing Weights

As enumerated in Table 5-4, three sets of routing weights were used in the study with relative emphasis on starting time, service duration, and path length. Most of the studies were performed with the emphasis on service duration (W2).

Table 5-4: Routing Weight Parameters

Weight Emphasis	Alpha	Beta	Gamma
W1: Path Length	0.1	0.1	0.8
W2: Duration	0.3	0.6	0.1
W3: Starting Time	0.6	0.3	0.1

5.2.3.2 Start Time Window

The allowable starting window controls the interval where the LSP may begin service. Two parameter variations were considered during the study:

Normal: $\pm$ 10 grid time units corresponding to 40% of mean duration of 25,

Expanded: $\pm$ 100 grid time units corresponding to 4 times the mean duration of 25.

The use of the expanded window variation allows for the analysis of the effect of window size on network performance and CPU processing.

5.3 Traffic Rate Sensitivity for FP and SP Routing

5.3.1 Overview

The traffic sensitivity assesses the effect of traffic rate on the relative performance of fixed path routing versus switched path routing for both lambda and packet switching. This

comparison is based on blocking, service duration, path length, start time deviation, and LSP CPU time metrics for scenarios based on moderate, heavy, and very heavy traffic rate loads on a 15 node network. The test parameters are summarized in the following table.

Table 5-5: Traffic Sensitivity Parameters

a.) Inputs		b.) Outputs
Fixed and Switched Path Routing		Number of Successful LSPs
Lambda and Packet Switching		LSP Path Length, Network Utilization (TE Link Average BW or λ Utilization)
Weights: W2		Duration Start Time Deviation
15 Node		% Multi-Segment Paths
Traffic Rates R1 – Very Heavy Load R2 – Heavy Load R3 – Moderate Load		CPU Time Average / LSP Average /Successful LSP Average /Failed LSP

Figure 5-2 depicts the relative performance of Fixed Path (FP) and Switched Path (SP) routing as the traffic rate changes for both Lambda and Packet Switching using the 15 node network.

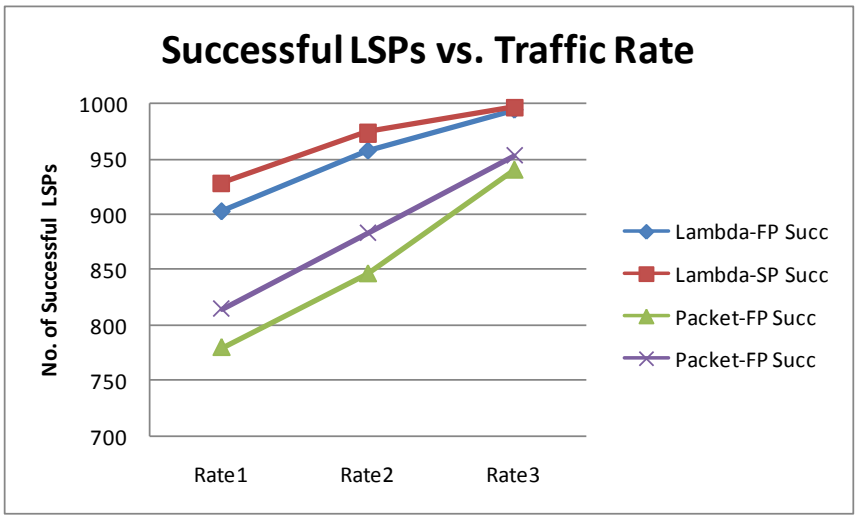

Figure 5-2: Traffic Sensitivity Performance – Successful LSPs vs. Traffic Rate

As shown in the figure, the performance of FP and SP is comparable, but in the heavier traffic cases (Rate1 and Rate2), SP routing provides a modest, but perceptible improvement. As shown in the figure, the improvement is more noticeable for packet switching (lower curves) than lambda switching (upper curves).

The following sections provide more detail on the results for packet switching and lambda switching technologies.

5.3.2 *Packet Switching Technology*

Tables 5-6 and 5-7 present the detailed network statistics comparing FP and SP routing for Packet Switching showing:

number of LSP attempts (1000 in all cases) and successes in the simulation,
average LSP service duration) ,
service LSP duration weighted by LSP data rate,
average path length (number of TE links),
average network utilization (averaged over all TE links), and
start time deviation (delta start) equal to the average difference between the requested start time and actual start time.

The comparison of FP versus SP routing is a complex multi-dimensional problem. First, SP routing successfully completes slightly more LSPs than FP routing (4% at the higher Rate1 and 1% the lower Rate3). However, SP routing also tends to improve start time, duration, and path length.

Packet		Packet-FP	LSP Total	LSP Success	Ave Duration	AveRW Duration	Ave Path Length	Ave Net Utilzn	Delta Start
FP		Rate1	1000	780	22.83	113.50	2.92	4.64	1.87
		Rate2	1000	846	23.23	119.69	2.90	4.09	1.53
		Rate3	1000	940	23.64	126.62	2.79	3.17	1.00

Table 5-6: Packet Network Performance – FP

Packet		Packet-SP	LSP Total	LSP Success	Ave Duration	AveRW Duration	Ave Path Length	Ave Net Utilzn	Delta Start
SP		Rate1	1000	815	22.63	114.73	2.61	4.39	1.65
		Rate2	1000	883	23.17	120.92	2.56	3.76	1.35
		Rate3	1000	953	23.70	127.63	2.51	2.87	0.81

Table 5-7: Packet Network Performance – SP

These results are captured over a period where the LSP loading varies over the mean steady-state value. Figure 5-3 depicts a representative case for the number of active LSPs versus simulation time for the Rate1 traffic loading. For Rate2 and Rate3, the traffic loading is similar, but spread out over proportionately longer periods of time.

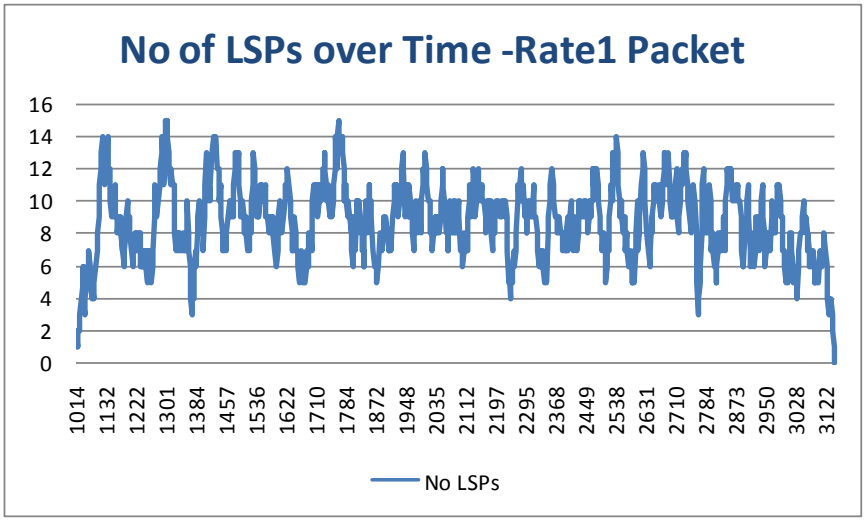

Figure 5-3: Packet LSP Loading – FP Routing

Figure 5-4 depicts the relative performance of FP and SP routing for the duration, start time, and path length metrics. SP routing provides uniform improvement except for the Duration Rate3 data point where there is negligible difference.

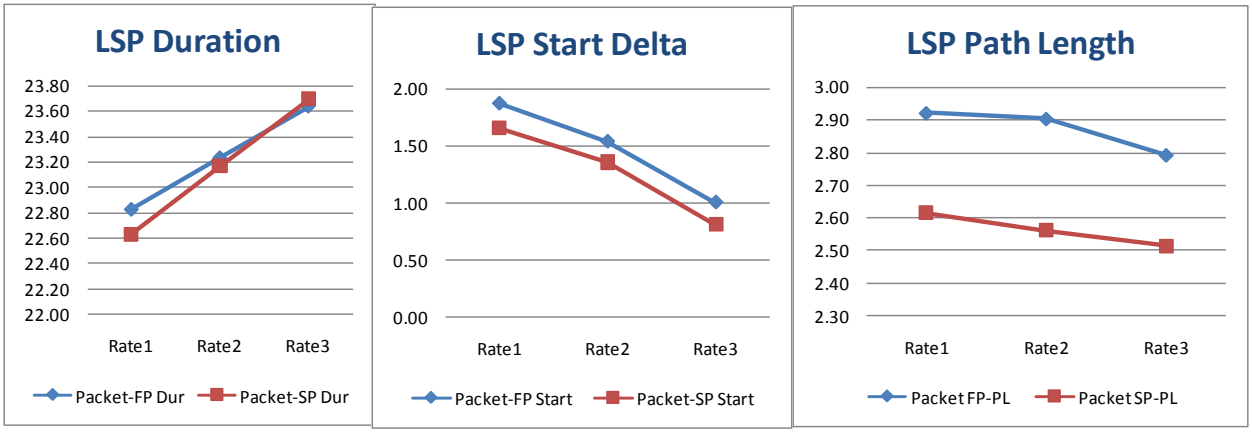

Figure 5-4: Packet Switching Technology Network Graphic Comparison – 15 Nodes

As mentioned in the previous section, the average requested service duration is 25 grid units. Therefore, as shown in the duration graphic in Figure 5-4, FP and SP routing provide approximately 90-95% of the requested service duration of 25 grid units depending upon the traffic rate.

From the figure, the FP and SP results show the least variation for the duration metric and the most variation for the path length metric. This is not surprising since the weights placed the most emphasis on duration; the values used for this test case were

Start time, $\alpha = 0.3$,
Duration, $\beta = 0.6$,
Path length $\gamma = 0.1$.

For the starting time metric, SP routing does provide a uniform improvement over FP routing of approximately 0.2 grid time units or approximately 2% of the start time window size. While this occurred consistently, this by itself is not especially important.

For the path length metric, SP routing does generate paths for LSPs that are on average 12% shorter than paths generated by FP routing. This advantage may allow SP routed networks to carry more traffic.

Since SP routing allows multiple paths to be used during the service duration, statistics were collected to analyze the frequency of LSPs using multiple paths, referred to as Multi-Segment LSPs. As shown in Table 5-8 for packet switching, Multi-Segment LSPs occur most frequently for the highest traffic rate (R1) as expected. In this case nearly 50% of the LSPs use multiple paths (389/815 = 47.7%) and there are 1.84 segments (paths) per LSP. For the lowest rate (R3), the number of multi-segment LSPs is reduced to 36% (347/953) and there are only 1.58 segments per LSP. For multi-segment LSPs, the average number of segments ranged from 2.85 to 3.33 for Rates 1 and 3.

Table 5-8: Switched Path Segment Statistics – Packet Switching

	Success LSPs	Total Segments	Segments per LSP	Multi Seg LSPs	Segments per MS LSP
Rate1	815	1498	1.84	389	2.85
Rate2	883	1509	1.71	377	3.01
Rate3	953	1501	1.58	347	3.33

Table 5-9 presents the CPU time per LSP averaged over all LSPs, successful LSPs, and failed LSPs for both SP and FP using packet switching averaged over 1000 samples. The major result derived from this data is that SP requires more processing than FP, and CPU processing time increases as the traffic rate decreases (Rate1>Rate2>Rate3).

However, the LSP CPU time is relatively flat as a function of traffic rate for a fixed size network – indicating that both the SP and FP algorithms are scalable relative to traffic. The arrival rates modeled were on the order of 1-2 service requests per grid time, and grid times are expected to be measured in hours, days, or weeks. Therefore, the sub-second CPU processing load per LSP with these algorithms is relatively modest indicating that the algorithms are scalable.

Table 5-9: LSP CPU Times (MSec) for Packet Switching – 1000 Samples

FP	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP		SP	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP
Rate1	14.75	14.51	15.63		1000	22.84	22.29	25.31
Rate2	17.73	17.50	19.00		1000	26.32	25.98	28.81
Rate3	22.15	21.99	24.32		1000	31.03	30.96	32.14

The processing becomes more intense as the system becomes more heavily loaded. Table 5-10 shows the CPU times for LSP service request 801 to 950. These processing times are approximately double the times averaged over the entire simulation, but show the same trends regarding FP versus SP performance and traffic rate trends.

Table 5-10: LSP CPU Times (MSec) for Packet Switching – Peak Period

FP-Peak	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP		SP-Peak	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP
Rate1	29.73	29.64	30.02		150	47.69	47.52	48.44
Rate2	37.09	36.96	37.74		150	56.49	56.36	57.26
Rate3	47.58	47.54	48.94		150	56.02	56.03	56.09

5.3.3 Lambda Switching Technology

Tables 5-11 and 5-11 present the detailed network performance statistics comparing FP and SP for Lambda Switching, respectively.

Table 5-11: Lambda Network Performance – FP

Lambda-FP	LSP Total	LSP Success	Ave Duration	Ave Path Length	Ave Net Utilzn	Delta Start
Rate1	1000	903	23.13	2.89	1.09	1.505
Rate2	1000	958	23.88	2.85	0.90	0.903
Rate3	1000	995	24.35	2.72	0.62	0.248

Table 5-12: Lambda Network Performance – SP

Lambda-SP	LSP Total	LSP Success	Ave Duration	Ave Path Length	Ave Net Utilzn	Delta Start
Rate1	1000	928	23.29	2.61	1.01	1.256
Rate2	1000	974	23.92	2.56	0.82	0.653
Rate3	1000	997	24.33	2.51	0.57	0.217

These results are captured over a period where the LSP loading varies over the mean steady-state value. Figure 5-5 depicts a representative case for the number of active LSPs versus simulation time for the Rate1 traffic loading.

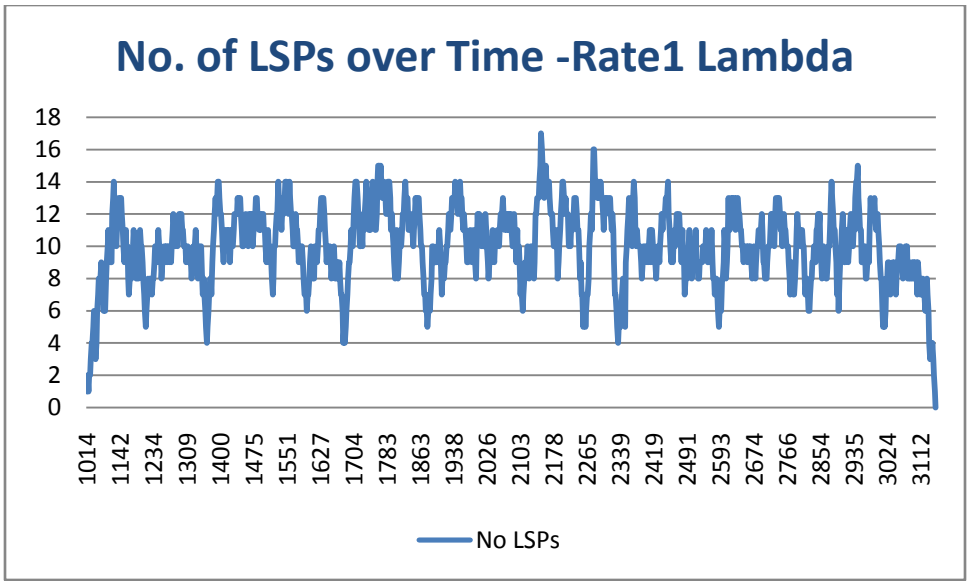

Figure 5-5: Lambda LSP Loading

The results are analogous to packet switching in that SP provides only a modest improvement over FP. Figure 5-6 depicts the LSP duration, start time, and path length performance.

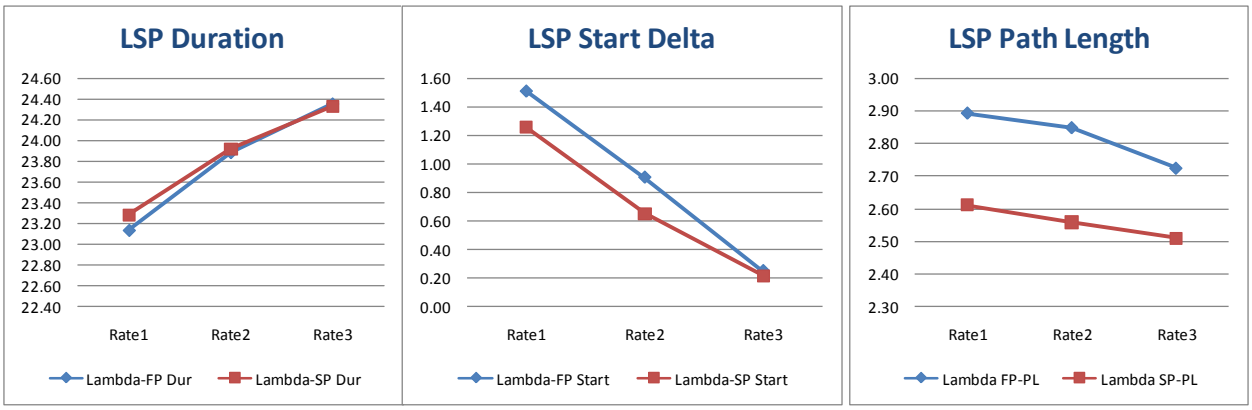

Figure 5-6: Lambda Switching Technology Network Graphic Comparison – 15 Nodes

The network performance results for Lambda Switching follow the same trends as the Packet Switching results described above. SP routing provides modes improvements in the number of LSPs successfully completed (3% at Rate1 and <1% at Rate3). For the duration, start time, and path length metrics, SP provides uniform improvements.

For duration, SP closely tracks the FP performance because the duration metric has the highest weighting while there is a more noticeable improvement in path length (approximately 10%). The improvement in start time is modest and only occurs and the heavier traffic rates.

Table 5-13 presents the CPU time per LSP averaged over all LSPs, successful LSPs, and failed LSPs for both SP and FP using lambda switching for 1000 samples. It follows the same pattern as described above for packet switching:

- SP routing is more CPU intensive than FP routing,

- CPU processing increases as traffic decreases.

Table 5-13: LSP CPU Times (MSec) Using Lambda Switching – 1000 Samples

FP	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP		SP	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP
Rate1	13.20	13.00	14.99		1000	21.01	20.69	25.20
Rate2	15.35	15.29	16.80		1000	23.34	23.26	26.08
Rate3	17.94	17.91	19.75		1000	25.77	25.73	35.56

As discussed above for packet switching, the processing becomes more intense as the system becomes more heavily loaded. Table 5-14 shows the CPU times for LSP service request 801 to 950. These processing times are approximately double the times averaged over the entire simulation, but show the same trends regarding FP versus SP and traffic trends.

Table 5-14: LSP CPU Times (MSec) Using Lambda Switching – Peak Period

FP-Peak	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP		SP-Peak	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP
Rate1	24.75	24.73	24.82		150	40.70	40.74	40.41
Rate2	30.10	30.10	30.02		150	46.55	46.54	46.55
Rate3	36.52	36.54	26.12		150	52.65	52.66	38.78

5.4 Network Size Sensitivity for FP and SP Routing

5.4.1 Overview

The network size sensitivity assesses the effect of the number of network nodes on the comparison of fixed path routing versus switched path routing for both lambda and packet switching. This comparison is based on blocking, service duration, path length, start time deviation, and LSP CPU time metrics for scenarios based on 7 node, 15 node, and 32 node networks. The traffic rate was applied so that the loading intensity was approximately the same for independent of network size, i.e., proportional traffic rates. For this study, the 7, 15, and 32 node networks used rates Rate3 (moderate), Rate2 (heavy), and Rate1 (very heavy), respectively.

The test parameters are summarized in the following table.

Table 5-15: Scalability Parameters

a.) Inputs		b.) Outputs
Fixed and Switched Path Routing		Number of Successful LSPs
Lambda and Packet Switching		Average LSP Path Length, Network Utilization (TE Link Average BW or λ)
Weights: W2		LSP Duration Start Time Deviation
7, 15, and 32 Nodes		
Traffic Rates		CPU Time

R1 for 32 Node Network R2 for 15 Node Network R3 for 7 Node Network		Average / LSP Average/ Successful LSP Average /Failed LSP

Figure 5-7 presents the number of successful LSPs for FP and SP for Packet and Lambda switching technologies for 7, 15, and 32 node network cases using proportional traffic rates to enable a fair comparison. For each case, the number of successful LSPs is relatively flat, i.e. within 2.5% of the average over 7, 15, and 32 node data points. However, the simulation time for the smaller network is proportionately longer to carry the same number of LSPs.

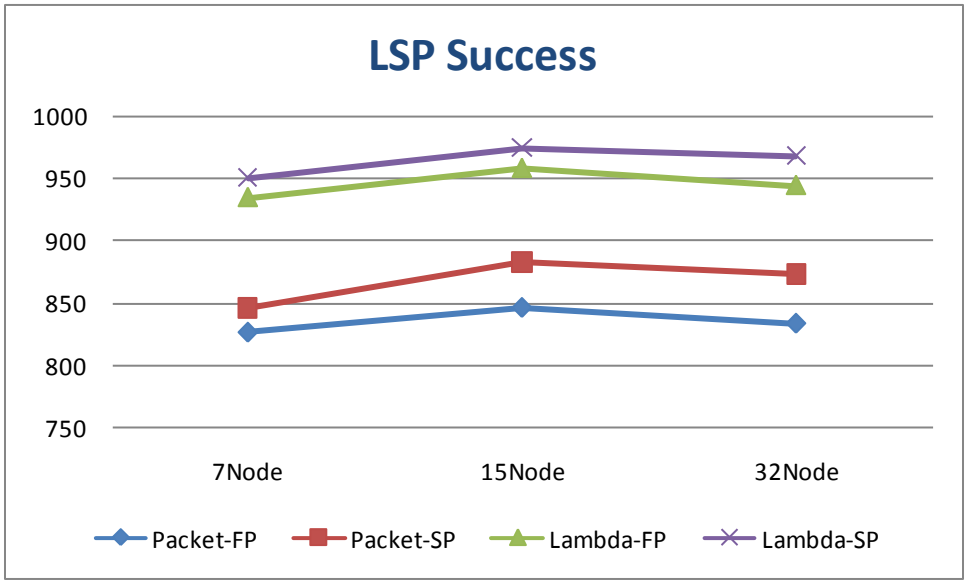

Figure 5-7: LSP Success Total with Proportional Traffic Rate

5.4.2 Packet Switching Technology

This section addresses the scalability of packet switching technology using the proportional rate traffic loading. Tables 5-16 and 5-17 present the network performance statistics comparing the 7, 15, and 32 node cases for FP and SP routing, respectively, using the same metrics defined in Section 5.3 for the 15 node case.

Table 5-16: Packet Switching Network Performance Statistics 7-15-32 Node Scalability – FP

Packet-FP	LSP Total	LSP Success	Ave Duration	AveRW Duration	Ave Path Length	Ave Net Utilzn	Delta Start
7 Node	1000	827	23.08	116.24	1.88	4.42	1.63
15Node	1000	846	23.23	119.69	2.90	4.09	1.53
32Node	1000	834	23.11	117.76	4.40	3.62	1.61

Table 5-17: Packet Switching Network Performance Statistics 7-15-32 Node Scalability – SP

Packet-SP	LSP Total	LSP Success	Ave Duration	AveRW Duration	Ave Path Length	Ave Net Utilzn	Delta Start
7Node	1000	846	23.23	117.55	1.74	4.22	1.53
15Node	1000	883	23.17	120.92	2.56	3.76	1.35
32Node	1000	873	22.66	117.17	3.88	3.27	1.32

Figure 5-8 depicts the performance of the duration, start time, and path length component of the objective function for packet switching.

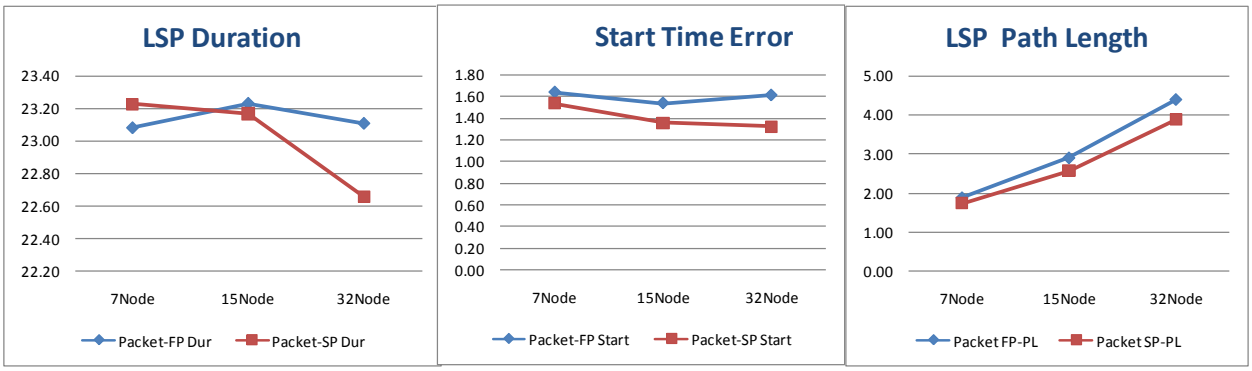

Figure 5-8: LSP Packet Switching 7-15-32 Node Network Comparison

SP routing tends to provide better performance than FP routing, e.g., smaller start time error and shorter paths. In the 32 node case, FP routing generated LSPs with a slightly longer duration (23.11 grid units versus 22.66 grid units), but in this case SP routing completed 4.7% more LSPs (873 versus 834)

Table 5-18 presents the CPU performance statistics for the 7, 15, and 32 node cases for both FP and SP routing in terms of the metrics described in Section 3 for the 15 node case using 1000 samples. Then Figure 5-9 graphically presents the CPU processing time per LSP as a function of network size for both FP and SP routing.

Table 5-18: Packet Switching CPU Performance Statistics (Msec) for 7-15-32 Nodes – 1000 Samples

Packet	Fixed Path				Switched Path				SP/FP Ratio		
	CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP		CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP		CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP
7Node	10.51	10.51	10.46		15.01	14.99	15.07		1.43	1.43	1.44
15Node	17.75	17.50	19.00		26.33	25.98	28.81		1.48	1.49	1.52
32Node	33.45	34.09	30.18		46.02	46.06	45.83		1.38	1.35	1.52

As the network increases in size from 7 nodes to 32 nodes, the number of nodes increases by a factor of 4+ and the number of TE links increases by a factor of 5+ (10 to 56). However, the CPU processing time only increases by a factor of 3.3 for FP (10.51->33.45ms) and 3.1 for SP (15.01->46.02 ms). From these results, these algorithms appear to scale well.

In all cases, SP processing times are longer than FP processing times with a ratio ranging from 1.38 to 1.48). Also, the CPU processing time is relatively independent of whether the LSP set up is successful or not.

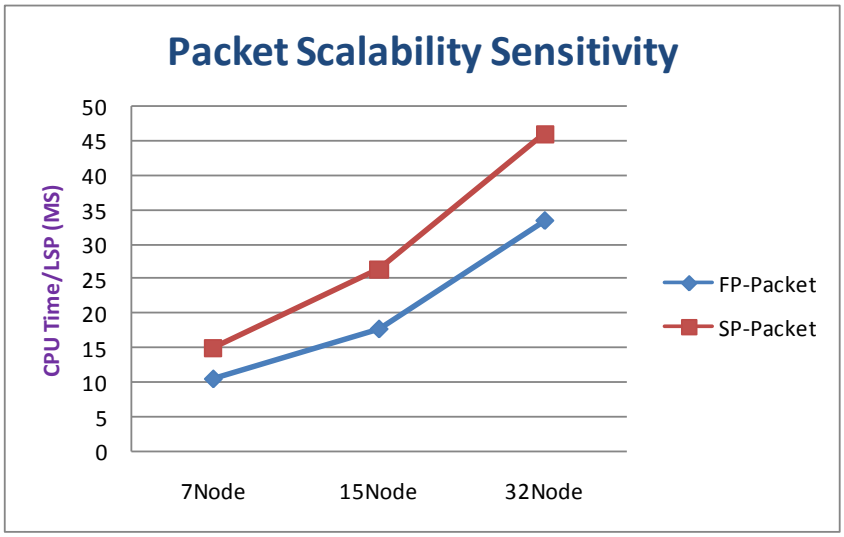

Figure 5-9: Packet Switching Performance Scalability Graphic FP vs. SP -1000 Samples

Table 5-19 shows the CPU loadings for the LSP service requests 801 to 950 while Figure 5-10 depicts the results in graphical format. It shows the same scalability and FP versus SP trends as the 1000 sample results.

Table 5-19: Packet Switching CPU performance Statistics (Msec) for 7-15-32 Nodes – Peak Loading

Packet	Fixed Path-Peak				Switched Path-Peak				SP/FP Ratio		
	CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP		CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP		CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP
7Node	22.56	22.53	22.72		31.50	31.47	31.70		1.40	1.40	1.40
15Node	37.09	36.96	37.74		56.49	56.36	57.26		1.52	1.52	1.52
32Node	62.98	63.18	61.93		97.61	97.66	97.16		1.55	1.55	1.57

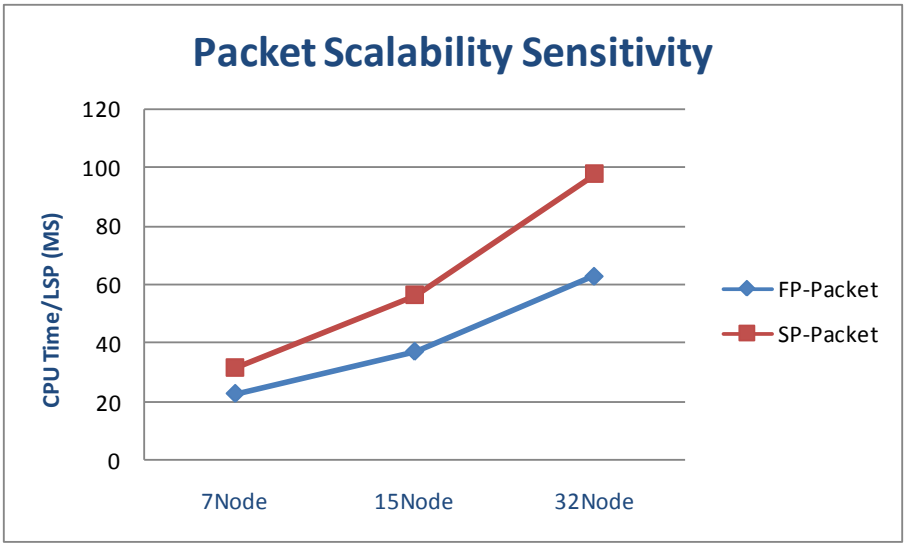

Figure 5-10: Packet Switching Performance Scalability Graphic FP vs. SP –Peak

5.4.3 Lambda Switching Technology

This section addresses the scalability of Lambda Switching technology. Tables 5-20 and 5-21 present the network performance statistics comparing the 7, 15, and 32 node cases for FP and SP routing, respectively, using the same metrics defined in Section 5.3.

Table 5-20: Lambda Switching Network Performance Statistics 7-15-32 Node Scalability –FP

Lambda-FP	LSP Total	LSP Success	Ave Duration	Ave Path Length	Ave Net Utilzn	Delta Start
7 Node	1000	935	23.52	1.87	0.99	1.06
15Node	1000	958	23.88	2.85	0.90	0.90
32Node	1000	944	23.55	4.40	0.82	1.03

Table 5-21: Lambda Switching Network Performance Statistics 7-15-32 node Scalability - SP

Lambda-SP	LSP Total	LSP Success	Ave Duration	Ave Path Length	Ave Net Utilzn	Delta Start
7Node	1000	950	23.69	1.76	0.95	0.96
15Node	1000	974	23.92	2.56	0.82	0.65
32Node	1000	967	23.34	3.92	0.73	0.86

Figure 5-11 depicts the performance of the duration, start time, and path length component of the objective function for lambda switching. SP routing tends to provide better performance than FP routing except for the duration metric for the 32 node case. In this instance, SP provides much better in the other metrics.

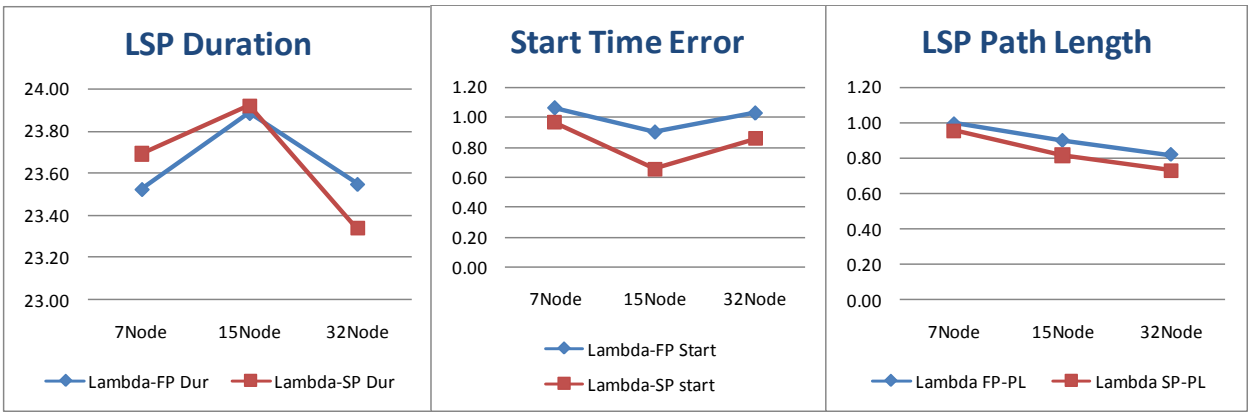

Figure 5-11: Lambda Switching 7-15-32 Node Network Comparison

Table 5-22 presents the CPU performance statistics for the 7, 15, and 32 node cases for both FP and SP routing in terms of the metrics described above for Lambda switching using 1000 samples. Then Figure 5-12 graphically presents the CPU processing time per LSP as a function of network size for both FP and SP routing.

Table 5-22: Lambda Switching CPU Performance Statistics (MSec) 7-15-32 Nodes – 1000 Samples

Lambda	Fixed Path				Switched Path				SP/FP Ratio		
	CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP		CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP		CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP
7Node	9.12	9.08	9.53		14.29	14.27	14.07		1.57	1.57	1.48
15Node	15.44	15.29	16.80		23.38	23.26	26.08		1.51	1.52	1.55
32Node	30.46	30.45	30.74		40.84	40.74	42.80		1.34	1.34	1.39

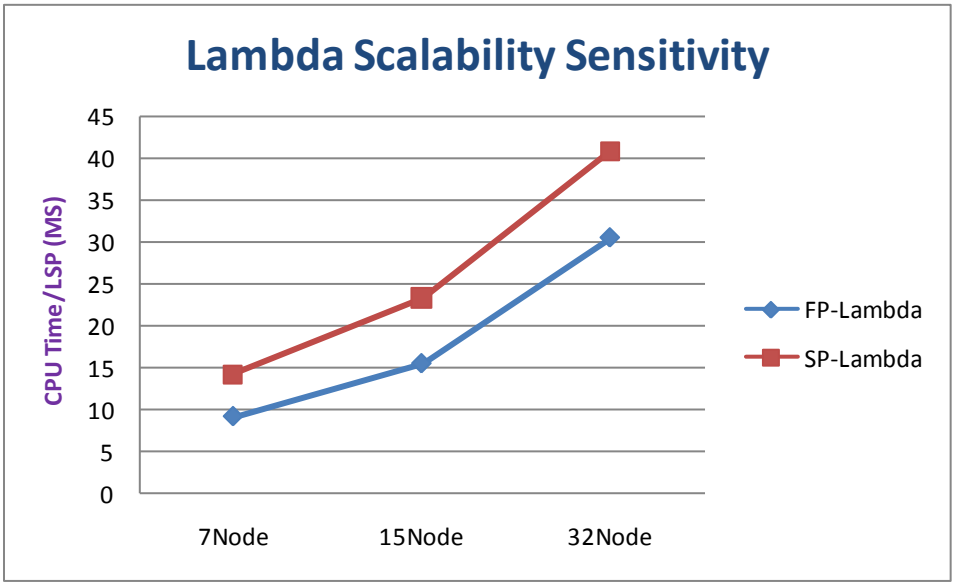

Figure 5-12: Lambda Switching Performance Scalability Graphic FP vs. SP – 1000 Samples

As the network increases in size from 7 nodes to 32 nodes, the number of nodes increases by a factor of 4+ and the number of TE links increases by a factor of 5+ (10 to 56). However, the CPU processing time only increases by a factor of 3.3 for FP (9.12->30.46ms) and 2.9 for SP (14.29->40.84 ms). From these results, these algorithms appear to scale well.

In all cases, SP processing times are longer than FP processing times with a ratio ranging from 1.34 to 1.57). Also, the CPU processing time is relatively independent of whether the LSP set up is successful or not for Lambda switching.

Table 5-23 depicts the corresponding statistics for service requests 801 to 950 while Figure 5-13 presents the results in graphical form. It shows the same trends as the 1000 sample results.

Table 5-23: Lambda Switching CPU Performance Statistics (MSec) 7-15-32 Nodes – Peak

Lambda	Fixed Path				Switched Path				SP/FP Ratio		
	CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP		CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP		CPU Time/LSP	CPU Time /Success LSP	CPU Time /Failed LSP
7Node	18.26	18.20	18.96		27.13	27.08	27.50		1.49	1.49	1.45
15Node	30.10	30.10	30.02		46.55	46.54	46.55		1.55	1.55	1.55
32Node	45.48	45.69	42.82		81.43	81.41	82.65		1.79	1.78	1.93

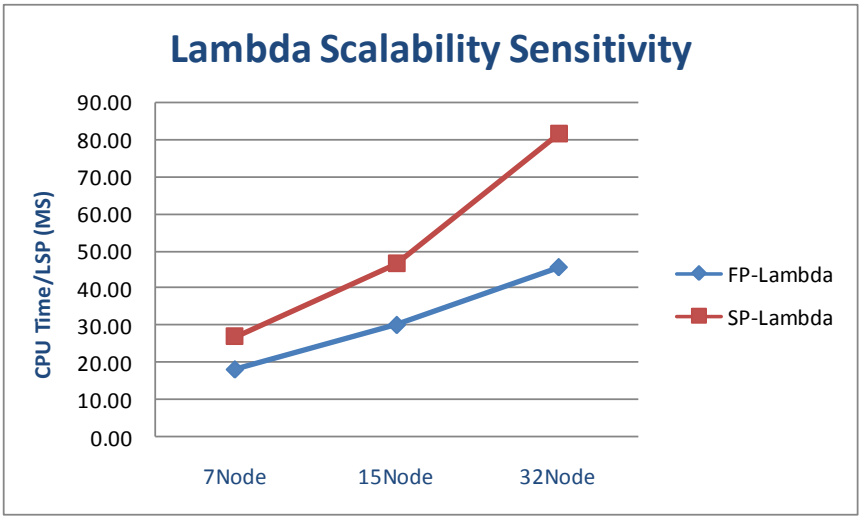

Figure 5-13: Lambda Switching Performance Scalability Graphic FP vs. SP – Peak

5.5 Routing Weight Sensitivity

The routing sensitivity assesses the effect of varying the objective function weights on the algorithm performance. As discussed above, these weights (α, β, γ) control the relative effect of starting time, duration, and path length has the objective function value, respectively. Table 5-24 enumerates the values used in this sensitivity study.

As part of this study, both packet and lambda switching technology were evaluated using both FP and SP routing with a start time window size of 10 grid time units. The results are presented in the following sections.

Table 5-24: Routing Weight Parameters

Weight Emphasis	Alpha	Beta	Gamma
W1: Path Length	0.1	0.1	0.8
W2: Duration	0.3	0.6	0.1
W3: Starting Time	0.6	0.3	0.1

5.5.1 Packet Switching Technology

Table 5-25 presents the routing weight sensitivity results for packet switching using both FP and SP routing. This table shows the results for each set of weights where W2 emphasizes duration, W1 emphasizes path length, and W3 emphasizes start time.

Table 5-25: Objective Function Weight Sensitivity – Packet Switching

Fixed Path	NTotal	Nsuccess	Ave Duration	AveRW Duration	AvePL	NetUave	Delta Start
W2-Duration	1000	780	22.83	113.50	2.92	4.64	1.87
W1-Path Length	1000	794	22.42	112.39	2.82	4.54	1.42
W3-Start Time	1000	783	22.24	110.50	2.92	4.54	1.28
Switched Path	**NTotal**	**Nsuccess**	**Ave Duration**	**AveRW Duration**	**AvePL**	**NetUave**	**Delta Start**
W2-Duration	1000	815	22.63	114.73	2.61	4.39	1.65
W1-Path Length	1000	815	22.63	112.54	2.68	4.36	1.48
W3-Start Time	1000	803	22.20	112.55	2.65	4.36	1.24

Figure 5-14 summarizes the conclusions of the analysis. Typically, the computed LSP service duration is relatively constant and is not significantly affected by either the weights or the routing while the success rate and computed path length are more affected by the routing. If a larger network were used, the path length may be more affected by the weights as well. However, the start time is affected by both the routing and the weights. Since the start time window is constrained to $\pm$ 10 grid time units around the desired start time, the reduction in start time deviation to 1.28 (FP) and 1.24 (SP) when the most weight is placed on start time appears significant.

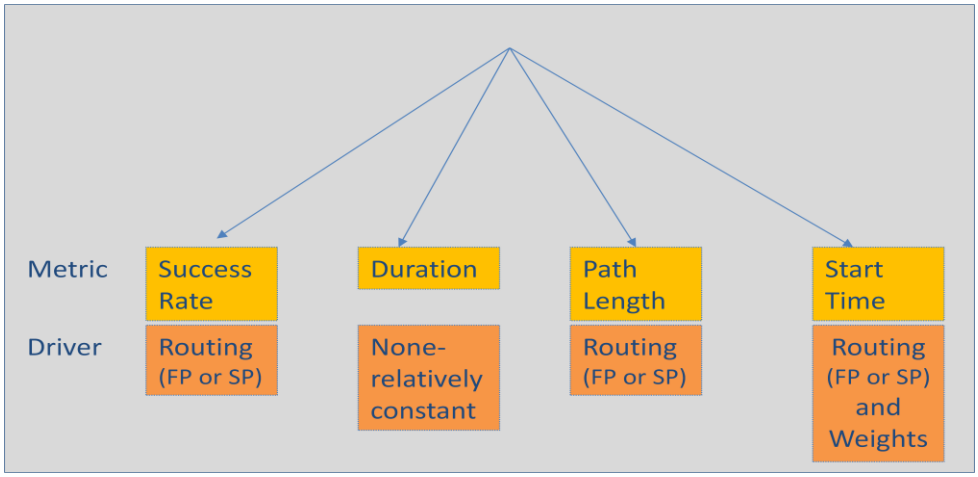

Figure 5-14: Impact of Objective Function Weights

5.5.2 *Lambda Switching Technology*

Table 5-26 presents the analogous routing weight sensitivity results for lambda switching using both FP and SP routing. These results follow the same trends as described above for packet switching where routing (FP or SP) has a bigger affect and the objective function weights (α, β, γ) have the biggest effect on the start time deviation metric.

Table 5-26: Routing Weights Sensitivity – Lambda Switching

Fixed Path	NTotal	Nsuccess	Ave Duration	AveRW Duration	AvePL	NetUave	Delta Start
W2-Duration	1000	903	23.13	23.13	2.89	1.09	1.51
W1-Path Length	1000	912	22.99	22.99	2.79	1.05	1.17
W3-Start Time	1000	911	22.79	22.79	2.80	1.05	0.98
Switched Path	**NTotal**	**Nsuccess**	**Ave Duration**	**AveRW Duration**	**AvePL**	**NetUave**	**Delta Start**
W2-Duration	1000	928	23.29	23.29	2.61	1.01	1.26
W1-Path Length	1000	928	23.17	23.17	2.62	1.00	1.15
W3-Start Time	1000	928	22.95	22.95	2.61	0.99	0.86

5.6 Start Time Window Sensitivity

5.6.1 *Overview*

The start time sensitivity assesses the effect of the allowing the start time to have a wider range on the comparison of fixed path routing versus switched path routing for packet switching. This comparison is based on blocking, service duration, path length, start time deviation, and LSP CPU time metrics for scenarios using window sizes of $\pm$10 grid units and $\pm$100 grid units with the 15 node network and heavy traffic rate R1. The test parameters are summarized in the following table.

Table 5-27: Expanded Window Parameters

a.) Inputs		b.) Outputs
Fixed and Switched Path Routing		Number of Successful LSPs
Packet Switching		LSP Duration Weighted LSP Duration Average Path Length Network Utilization Start Time Deviation
Traffic Rate: R1		CPU Time Average / LSP Average/ Successful LSP

		Average/ Failed LSP
15 Nodes		
W2		
Start Time Window: 10 and 100 Grid Units		

5.6.2 *Packet Switching Technology*

Tables 5-28 and 5-29 compare the network performance statistics for expanded window (100) with the original window (10) for FP and SP routing, respectively, using Packet switching. As indicated in the table, there is a major improvement in the number of successes (18.9% for FP and 17.0% for SP) as the window is increased from ±10 to ±100. Note the service duration decreased slightly in both cases even though the weights (W2) emphasize duration for the expanded window. Since the number of successful LSPs significantly increased, the network utilization (average BW per TE link) also increased. However as expected, the average difference between actual start time and desired start time (delta start) increases by an order of magnitude because of the expanded window. For FP routing, delta start increased by a factor of 8.98 (1.87 to 18.53), and for SP routing delta start routing increased by a factor of 8.07 (1.65 to 14.98).

Table 5-28: Expanded Window Packet network Performance – Fixed Path

		NTotal	Nsuccess	Ave Duration	AveRW Duration	AvePL	NetUave	Delta Start
Fixed-Win100	Rate1	1000	927	22.55	119.61	3.03	5.79	18.53
Fixed-Win10	Rate1	1000	780	22.83	113.50	2.92	4.64	1.87
			18.90%	-1.22%	5.38%	3.61%	24.69%	8.93

Table 5-29: Expanded Window Packet Network Results – Switched Path

		NTotal	Nsuccess	Ave Duration	AveRW Duration	AvePL	NetUave	Delta Start
Switched-Win100	Rate1	1000	954	22.32	120.17	2.71	5.33	14.98
Switched-Win10	Rate1	1000	815	22.63	114.73	2.61	4.39	1.65
			17.01%	-1.35%	4.74%	3.79%	21.56%	8.07

With the expanded window, SP still provides only a modest (3%) improvement over FP (927 ->954) in the number of successful LSPs. However, the modest improvement in success rate and the larger improvement in path length may be significant in cases where the network is being finely tuned.

Table 5-30 compares the CPU time per LSP for expanded window (100) with the original window (10) for FP and SP respectively using 1000 Samples. These results show that the processing time for SP routing grows less quickly than the processing time for FP routing as the window size increases – FP increased by a factor of 3 while SP increased by a factor of 2. With the window size of 100, the processing times for FP and SP are roughly comparable, 39.76 and 42.25 msec, respectively. As shown in the tables, the SP processing is 50% greater with the window size of 10 (13.20 msec for FP and 21.01 msec for SP)

Table 5-30: Packet CPU Time (MSec) Results – 1000 Samples

	Fixed			Switched		
	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP
Win100	39.76	39.98	37.14	42.25	41.38	60.65
Win10	13.20	13.00	14.99	21.01	20.69	25.20
Ratio	3.01	3.08	2.48	2.01	2.00	2.41

Table 5-31 compares the CPU time per LSP for the peak loading using LSP service requests 801 to 950. Again the SP processing times grow less quickly than the FP processing times as the window size expands.

Table 5-31: CPU Time (MSec) Results - Peak

	Fixed			Switched		
	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP
Win100	Rate1	59.77	60.99	Rate1	73.22	72.43
Win10	Rate1	29.73	29.64	Rate1	47.69	47.52
Ratio	Ratio	2.01	2.06	Ratio	1.54	1.52

5.6.3 Lambda Switching Technology

Tables 5-32 and 5-33 compare the network performance for expanded window (100) with the original window (10) for FP and SP, respectively, using Lambda Switching. The results follow the same trends as described above for packet switching. However, the quantitative differences are somewhat less because the success rate in the window size 10 case is relatively higher leaving smaller margin for improvement.

For the Lambda case expanding the window from 10 to 100 results in nearly of the LSPs being set up successfully. As a result the delta start increases by a factor of 4.50 for FP and 3.40 for SP.

Table 5-32: Expanded Window Lambda Network Performance – Fixed Path

		NTotal	Nsuccess	Ave Duration	AvePL	NetUave	Delta Start
Fixed-Win100	Rate1	1000	996	23.47	2.90	1.18	8.28
Fixed-Win10	Rate1	1000	903	23.13	2.89	1.09	1.51
			10.34%	1.48%	0.38%	8.63%	4.50

Table 5-33: Window Lambda Network Performance – Switched Path

		NTotal	Nsuccess	Ave Duration	AvePL	NetUave	Delta Start
Switched-Win100	Rate1	1000	999	23.58	2.64	1.06	5.53
Switched-Win10	Rate1	1000	928	23.29	2.61	1.01	1.26
			7.58%	1.28%	0.95%	5.37%	3.40

Tables 5-34 compares the CPU time per LSP for expanded window (100) with the original window (10) for FP and SP respectively using 1000 samples. The processing times for both FP and SP increase as the window size increases, but the FP processing increases much more rapidly (3.37 for FP versus 1.74 for SP). As shown in the table, the FP processing (44.54 msec) is more intensive for the window size of 100 than that of SP (36.64 msec). Note that nearly all of the LSPs are successfully set up in this test case so the CPU times for failed LSPs re not statistically significant.

Table 5-34: Lambda CPU Time (MSec) Results– 1000 Samples

	Fixed			Switched		
	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP
Win100	44.54	44.52	15.91	36.64	36.59	25.41
Win10	13.20	13.00	14.99	21.01	20.69	25.20
Ratio	3.37	3.42	1.06	1.74	1.77	1.01

Table 5-35 compares the CPU time per LSP for the peak loading using LSP service requests 801 to 950. SP processing times grow less quickly than the FP processing times as the window size expands, but it still exceeds FP processing with the window size of 100.

Table 5-35: Lambda CPU Time (MSec) Results –Peak

	Fixed			Switched		
	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP
Win100	60.07	60.08	15.91	61.10	61.05	23.94
Win10	24.75	24.73	24.82	40.14	40.19	39.85
Ratio	2.43	2.43	0.64	1.52	1.52	0.60

5.7 Shortest Path Threshold Calculation Sensitivity

As described in Section 3, a major step in the optimization of the starting time, duration, and path involves the execution of a shortest path algorithm to compute as candidate paths. The shortest path threshold calculation sensitivity assesses alternative techniques for performing this shortest path calculation for switched path algorithms. Since the shortest path algorithm may be executed hundreds of times to perform this optimization, it is essential that this calculation be done efficiently.

From Section 3, the optimization algorithm requires aij(t) the shortest path between i and j for time t during the service interval and store the corresponding paths Pij. The most straight forward approach for computing the shortest path is an On Request approach where the Djikstra algorithm is invoked to compute aij(t) and Pij for each jump point during the service interval for each service request.

However, the network topology consisting of the TE links with available bandwidth changes infrequently. Therefore, an alternative approach is the Threshold approach to re-compute aij(t) only when the utilization on a TE link exceeds a specified threshold during some time interval (t1, t2). In this case, the aij(t) and Pij must be updated for all pairs that use the TE link whose threshold has been exceeded during (t1, t2).

The threshold based approach may be either:

- hard threshold where the threshold value is checked before assigning the current LSP or
- soft threshold where the threshold value is checked after the assigning the current LSP.

This comparison is applicable to switched path routing only because shortest paths are computed per jump point. Since FP routing computes shortest paths for each solution point, it would not be practical to pre-compute shortest paths in this case. Also, the threshold approach is not useful for lambda switching for all-optical networks because the availability of a single wavelength is critical rather than the total number of wavelengths that are available.

This comparison is based on blocking, service duration, path length, start time deviation, and LSP CPU time metrics for scenarios using the 15 node network with TE links having capacity of 10 Mbps and a threshold of 0.5. Packet LSPs require a bandwidth between 0 and 2 Mbps. The test parameters are summarized in Table 5-36.

Table 5-36: Threshold Sensitivity Parameters

a.) Inputs		b.) Outputs
Switched Path Routing		Number of Successful LSPs
Packet Switching		
Traffic Rates: R4		LSP Duration Weighted LSP Duration Average Path Length Network Utilization Start Time Deviation
15 Nodes		
W2		CPU Time Average / LSP Average Successful LSP Average Failed LSP
100 Time Unit Window		No. of Shortest Path Calculations Number of Algorithm Iterations

		Number of Solution Points Number of Jump Points
On-Demand and and Soft Threshold		Path Segments

Table 5-37 summarizes the network performance the SP algorithm with On Request shortest path calculation versus both the soft threshold algorithm. It presents the results in terms of number of successful LSPs, average duration, average path length, network utilization (average TE link utilization), and start time deviation. As shown in the table, the number of successful LSPs is comparable for each algorithm, but the soft threshold algorithm provides longer a duration, shorter path length, higher TE link utilization, and smaller start time deviation. This is expected because the soft algorithm allows the more bandwidth to be used on the TE links (because it does not enforce the threshold until after the LSP is assigned)..

Table 5-37: Network Performance Results

	NTotal	Nsuccess	Ave Duration	AveRW Duration	AvePL	NetUave	Delta Start
SP	1000	967	21.02	23.10	2.58	2.94	11.37
SP-Soft	1000	966	23.99	26.52	2.47	3.38	10.73

Table 5-38 summarizes the CPU performance the shortest path algorithm versus both the soft and hard threshold algorithms. Although the Soft algorithm shows a dramatic reduction in the number of shortest path calculations from the On-Request shortest path, the CPU time per LSP is comparable for the Soft Threshold and On Request Algorithms. However, the soft threshold algorithm requires on average an additional algorithm iteration (6.4 vs. 5.6) resulting in comparable CPU time. Therefore, the per iteration software overhead appears very significant.

Table 5-38: CPU Time Results – 1000 Samples

	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP	TimesOfSP_Alg	NumOfOptimAlg	NumOfSP	NumOfJP
SP	103.7	101.4	170.1	135.5	5.6	1547	1547
SP-Soft	115.0	112.3	191.0	47.3	6.4	1675	1675

Table 5-39 depicts the CPU results for the LSP service request 801 to 950 corresponding a peak loading resulting in a more intense CPU load – approximately 30% greater. So the On Request algorithm is still preferable.

Table 5-39: CPU Time Results - Peak

	CPU Time /LSP	CPU Time / Succ LSP	CPU Time / Fail LSP
SP	130.4	126.0	171.2
SP-Thresh	152.6	151.2	164.9

Because of the increased bandwidth available to the soft algorithm, the number of multi-segment paths is dramatically reduced with the Soft threshold algorithm. Table 5-40 shows the number of LSPs requiring two or segments for the On Request SP algorithm (492 LSPs) and SP with the Soft threshold algorithm (392 LSPs). Thus, the Soft threshold algorithm reduces the number of multi-segment LSPs by 100 LSPs over the On Request algorithm.

Table 5-40: Path Segment Results

	MS LSP
SP	492.33
SP-Soft	392.33

5.8 Conclusions

This section summarizes the conclusions for each sensitivity study described above in Sections 5.3 to 5.7.

5.8.1 Traffic Rate Sensitivity

For the traffic rate sensitivity study using the 15 node network, it was found that SP routing provided only modest improvements over FP routing. Using the W2 weight emphasizing duration in this study, FP and SP gave similar results for the duration metric, but SP provided uniform, but modest improvement for the other metrics for both packet and lambda switching technologies. In cases where the network is being finely tuned, the modest improvements in increased success rate, shorter path length, and start time deviation may justify the use of the more complex SP routing.

For SP routing using packet switching, the number of multi-segment paths increased as the traffic rate increased. At the highest traffic rate, R1, nearly 50% of the LSPs required multiple segments (389 out of 815 successful LSPs). In this case, the multi-segment LSP had on average 2.85 segments. This is a relatively modest impact given that the set up of LSPs is expected to be a relatively infrequent event. Typical parameter values might be:

- service times in the study were on average 25 grid units,
- grid times are measured in hours, days, or maybe weeks.

For example with grid times of days, paths would be re-routed approximately every week (25 days/3.33 = 7.5days) or less at lower traffic rates. Therefore, re-routing of SP paths does not occur very frequently and should not cause a significant degradation in performance.

For both packet and lambda switching, the CPU time per LSP is relatively flat as the traffic rate changes for both FP and SP. When the traffic rate increases introducing increasing network loading, there is no significant increase in CPU time – in fact the CPU time decreases.

While the CPU time per LSP is comparable for FP and SP, the FP time is slightly better despite requiring an increased number of shortest path calculations. However, SP increases less quickly than FP as the window size expands as discussed below in the window size sensitivity.

The same performance trends that occurred for the entire 1000 sample simulation were also observed in the peak period (samples 801 to 950).

5.8.2 *Network Size Sensitivity*

The network size sensitivity study analyzed the impact on the computational algorithms of increasing the network size from 7 nodes (10 TE links) to 15 nodes (26 TE links) to 32 nodes (56 TE links) for both packet switching and lambda switching technology. These tests were performed with a proportional traffic load such that the number of successful LSPs was approximately the same for each network size (but the simulation times were proportionately longer for the smaller networks). Also, the objective function weights, W2, were used placing the highest weight on duration and lowest weight on path length (α= 0.3, β=0.6, γ=0.1).

With these weights, FP and SP routing provided very similar duration metric results for both packet and lambda switching. However, SP routing was able to generate modest improvements in start time and path length metrics.

For these test cases, the algorithms show very nice scalability characteristics. As the network size increases by a factor of 4+, both the FP and the SP processing times increase less quickly for both packet and lambda switching. This trend was observed for both the average over the entire simulation as well as over the peak period (samples 801 to 950).

5.8.3 *Routing Weights Sensitivity*

The routing weight sensitivity study analyzed the responsiveness of the algorithms to changes to the objective function weights using the 15 node network for both packet and lambda switching. Three sets of weights were used:

Weight Emphasis	Alpha	Beta	Gamma
W1: Path Length	0.1	0.1	0.8
W2: Duration	0.3	0.6	0.1
W3: Starting Time	0.6	0.3	0.1

For the routing weights sensitivity study for packet switching with SP routing, it was found that there was a noticeable difference in the delta start time relative to the start time window as the weights placed more emphasis the start time metric. In particular, as more emphasis is place on start time, the start time deviation is reduced. However, for this network, the path length and duration show relatively small variations as these weights are emphasized.

Lambda switching demonstrated similar performance characteristics when the objective function weights were varied.

5.8.4 *Expanded Window Sensitivity*

For the expanded window sensitivity study for packet switching, it was found for both FP and SP routing that:

- number of successful LPS increased significantly (17-19%) when window increased from 10 to 100,
- path length is slightly longer (4%),
- TE link utilization is greater (24%),
- service duration is relatively flat.

The overall network performance for SP was only slightly better than for FP (approximately 3% improvement in LSP success rate). However, with the expanded window the start time deviation increases by a factor of 8.

However, the CPU time per LSP FP and SP with the expanded window is roughly comparable, 40 and 42 ms, respectively, over the entire simulation. With the windows size of 10, SP required 50% more CPU time. During the peak period, the disparity between FP and SP processing did not narrow as much. In this case SP required only 15% more processing.

For the expanded window sensitivity study for lambda switching, it was found for both FP and SP routing provide similar network statistical results. More interesting, the SP CPU times per LSP are less than the FP CPU times in the average case, 45 ms and 37 ms, respectively. In the peak case, the SP and FP CPU times are comparable, 61 and 60 ms, respectively. Thus, the SP may be preferable as the window size expands.

5.8.5 Shortest Path Calculation Sensitivity

The shortest path sensitivity study assesses an algorithm variation where shortest paths are pre-computed and then re-computed only when link utilization thresholds are exceeded. This option applies only for SP routing using packet switching. For this shortest path calculation alternative, it was found that:

- Soft threshold algorithm provides better network performance by reducing the number of multi-segment LSPs by 100 (because more bandwidth may be used since the threshold is enforced after the LSP is assigned),
- Soft threshold algorithm does not provide any CPU time improvements over the On Request algorithm even though the number of shortest path calculation is reduced significantly (135 -> 47) because more algorithm iterations are required,

Thus the On Request algorithm remains the preferred approach.

6 PHASE III PLANS

This section provides a summary of our plans for commercializing the distributed scheduling technology in Phase III. As shown in Figure 6-1, our ongoing fund raising and customer interaction efforts will continue to establish a basis of support for our efforts. After achieving this support, we will complete a networked PCE prototype and then enter the product development phase. Also with the support of our customers, we will standardize the scheduling enhancement to GMPLS and PCE protocols. Details of these plans are summarized below.

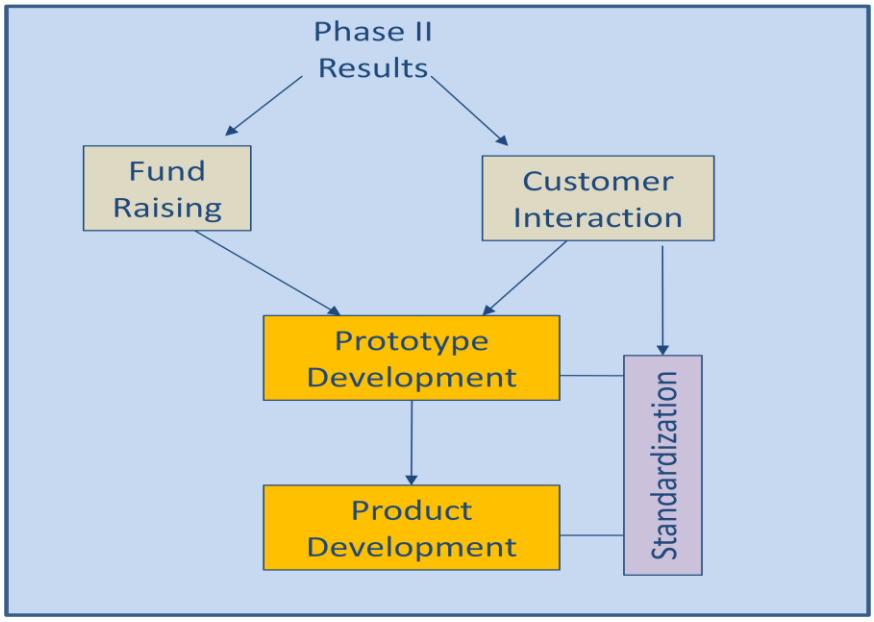

Figure 6-1: Phase III Plans

6.1 Prototype Development

In this activity we will develop a prototype networked PCE supporting an all-optical network, e.g., using the LN2000 as the switching node, in a single domain. It would support the scheduling of unprotected LSPs using the operations concept and architecture described in Section 3.

This prototype would have the PCC resident in the NMS and provide basic functionality to demonstrate proof of concept, e.g., fixed path LSPs. Since the prototype would be using GMPLS lambda switching with an all-optical platform, it would enforce the wavelength continuity constraint.

6.2 Product Development

After the proof of concept is demonstrated by the networked PCE prototype, we will begin product development. This will be an incremental process driven by our customer needs. The following sections summarize the features that are under consideration for implementation.

6.2.1 GMPLS and PCE Features

While the PCE working group has made major progress in the past year, it is an ongoing effort to achieve the maturity of the GMPLs standards. During our product development activity, we will implement the basic PCE standards and extend the protocols where necessary to support scheduling and develop associated algorithms. Figure 6-2 depicts potential activities that involve single domain recovery protocols (PCE algorithms), PCE discovery, PCE policy, multi-domain-protocols (PCE algorithms), multi-domain path confidentiality [], multi-domain recovery protocols (PCE algorithms), Re-optimization protocols (PCE algorithms), multi-layer protocols (PCE Algorithms), and multi-layer recovery protocols (PCE algorithms).

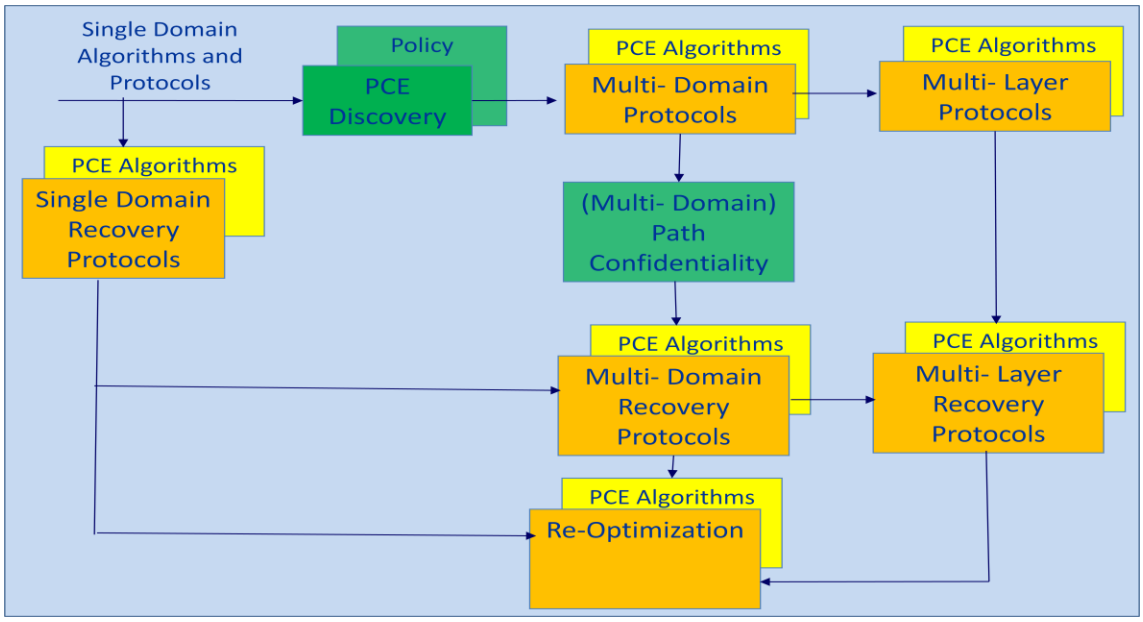

Figure 6-2: Phase III PCE-GMPLS Roadmap

In extending the GMPLS protocols support scheduling, we will address the use of both FP and SP routing. The use of SP may be more computational tractable for some applicable technologies (Ethernet) in some cases, e.g., inter-domain applications.

Other issues may have to be addressed. For example in case of a control plane failure, it may be necessary for a node to retrieve the schedule from its neighbors, i.e., similar to what is currently done in a graceful restart for existing LSPs.

Also, GMPLS defines many switching technologies. In order to be responsive to customer needs, it will be necessary additional multiple technologies beyond lambda supported in the prototype. Ethernet is a very a high priority.

6.2.2 Computational Features

The computational capabilities of the PCE will have to be enhanced during the product development phase. Two immediate enhancements are envisioned:

- Batch services – groups of LSPs are serviced together such that they may be jointly optimized
- Stateless PCE – allowing support of very large numbers of LSPs typical of Ethernet networks

Also, the use of load-sharing PCEs will also be investigated.

6.2.3 Operational Features

In order to facilitate deployment of distributed scheduling, it will be necessary to enhance the software to provide additional capability. Such capabilities may include nodal redundancy or a web services interface.

6.3 IETF Standardization

We envision working with our customers to standardize the scheduling enhancements to GMPLS and PCE protocols. After we demonstrate the proof of concept of the distributed

scheduling technology, we intend work with our customers to prepare draft documents for submission to the IETF working groups on GMPLS and PCE.

6.4 Customer Interation

We will continue our ongoing interaction with telecommunications carriers in the United States, Europe, and Asia to understand their scheduled applications, use their facilities for prototype demonstration, solicit their help in protocol standardization, and identify sales opportunities.

6.5 Fund Raising

We have been in contact with venture capitalist to fund Phase III and will follow-up on these contacts.

7 REFERENCES

7.1 General

[1.] P. Torab., "Dynamic path scheduling through extensions to Generalized Multiprotocol Label Switching (GMPLS)," Phase I for the US Department of Energy SBIR Grant Number DE-FG02-06ER84515, April 2007.

[2.] M. Mohri, "Semiring frameworks and algorithms for shortest-distance problems," Journal of Automata, Languages and Combinatorics, vol. 7, no. 3, March 2002, pp. 321-350.

[3.] M. Gondran and M. Minoux, Graphs and Algorithms, Wiley, New York, 1984.

[4.] J. Y. Yen, "Finding the k shortest loopless paths in a network," Management Science, vol. 17, no. 11, July 1971, pp. 712-716.

[5.] E. L. Lawler, "A procedure for computing the k best solutions to discrete optimization problems and its application to the shortest path problem," Management Science, vol. 18, no. 7, March 1972, pp. 401-405.

7.2 Normative References

7.2.1 Architecture

[6.] ITU-T publication G.8080, *Architecture for the Automatically Switched Optical Network (ASON)*, October 2001.

7.2.2 GMPLS Routing, OSPF

[7.] R. Coltun, "The OSPF Opaque LSA Option," RFC 2370, July 1998.

[8.] D. Katz, K. Kompella and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2," RFC 3630, September 2003.

[9.] K. Kompella et.al., Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", RFC 4202, October 2005

[10.] K. Kompella and Y. Rekhter, "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)," RFC 4203, October 2005.

[11.] IANA OSPF Traffic Engineering TLVs registry at http://www.iana.org/assignments/ospf-traffic-eng-tlvs.

[12.] K. Ishiguro et. al., "Traffic Engineering Extensions for OSPF Version 3", RFC 5329, September 2008.

[13.] R. Coltun et. al., "OSPF for IPv6," RFC 5340, July 2008.

7.2.3 GMPLS Signaling, RSVP

[14.] R. Braden et al., "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification," RFC 2205, September 1997.

[15.] D. Awduche et al., "RSVP-TE: Extensions to RSVP for LSP Tunnels," RFC 3209, December 2001.

[16.] B. Jamoussi et al., "Constraint-Based LSP Setup using LDP," RFC 3212, January 2002.

[17.] L. Andersson et al., "The Multiprotocol Label Switching (MPLS) Working Group decision on MPLS signaling protocols," RFC 3468, February 2003.

[18.] L. Berger et al., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description," RFC 3471, January 2003.

[19.] L. Berger et al., "Generalized Multi-Protocol Label Switching (GMPLS) Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions," RFC 3473, January 2003.

[20.] IANA RSVP parameters registry at http://www.iana.org/assignments/rsvp-parameters.

[21.] D. Papadimitriou, "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Extensions for G.709 Optical Transport Networks control," RFC 4328, January 2006.

7.2.4 PCE

[22.] A Farrel, J.P. Vasseur, and G. Ash, "A Path Computation Element (PCE)," RFC 4655, August 2006.

[23.] J.P. Vasseur et. Al., Path Computation Element (Communication Protocol," RFC 5440, March 2009.

[24.] I. Boyskin et. Al., "Policy Enabled Path Computation Framework," RFC 5394," December 2008.

[25.] J.L. LeRoux et. Al., "OSPF Extensions for Path Computation Element (PCE) Discovery", RFC 5088, September 2007.

[26.] E Oki et.al., "Framework for PCE-Based Inter-Layer MPLS and GMPLS Traffic Engineering, draft,", June 2008.

[27.] E. Stephan, "Definition of Managed Objects for Path Computation Element Discovery – draft," July 2008.

[28.] E. Oki, " Extensions to the Path Computation Element Communication Protocol (PCEP) for Route Exclusion," RFC 5521, April 2009.

7.2.5 Other

[29.] D. L. Mills, "Network Time Protocol (Version 3) Specification, Implementation and Analysis," RFC 1305, March 1992.