Optimal Core–Edge Storage Area Network Design

Timothy David Thompson
Supervised by Dr. M. O'Sullivan and Dr. C. Walker
Department of Engineering Science
University of Auckland, New Zealand
ttho035@ec.auckland.ac.nz

Abstract

Storage Area Networks (SANs) have become a unique solution for information management as the integrity and reliability of data storage become increasingly important within industry. A SAN manages the transfer of data from sets of servers and/or clients to centralized data storage through an internal fabric comprising of switches, hubs and links.

Core-Edge is a reference topology widely used in the layout of SAN fabric. We study the problem of mapping this SAN fabric for specific demands, utilizing Core-Edge design. We present an integer programming (IP) model for this restricted network design problem and offer preliminary results and discussion on standards for Core-Edge SAN fabric design.

1 Introduction

Data management has developed into an important issue within industry as a consequence of increased dependency on electronic storage. With concerns for the security, reliability and availability of data within large-scale networks, companies are increasingly turning to centralized storage as a solution. Centralized storage permits data management to be externalized from the local computer network. This allows for better overall resource allocation and a reduction in complexity for administration and management of data. Multi-user access to the centralized storage is attained through the use of Storage Area Networks (SANs).

A SAN is a high-speed sub-network, secondary to a Local Area Network (LAN) or Wide Area Network (WAN). SANs manage the transfer of data from sets of servers and/or clients (*hosts*) to the centralized storage. The centralized storage incorporates media such as disks, disk arrays and tape drives (*devices*). The hosts and devices are connected through the internal fabric of the SAN comprising of

switches, *hubs* and *links*. Figure 1 illustrates the relationship between a SAN and LAN to connect each workstation to all attached devices. The arrangement of the LAN and SAN fabric is customized for the bandwidth requirements across both sub-networks.

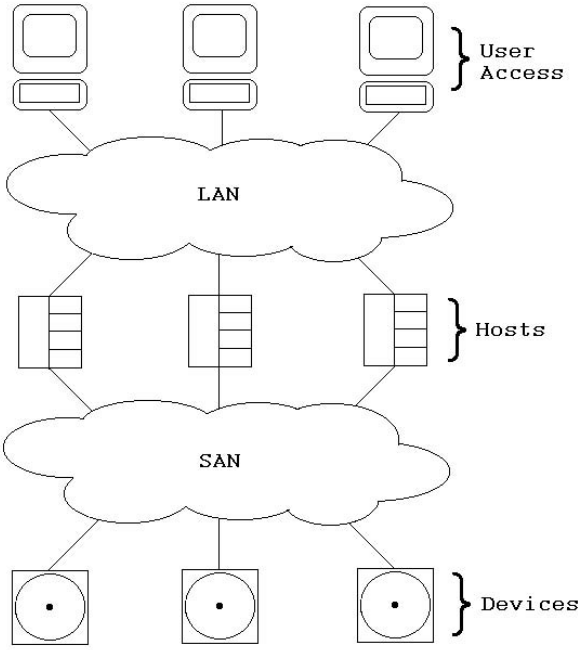

Figure 1: Generalized network featuring a SAN

Industry shows that 10-20% of the total storage system cost can be attributed to the design and installation of the SAN (Ward et al. 2002). IT experts traditionally spend considerable time and effort manually designing SAN fabric for even small applications. Currently SANs can reach a size of 50 hosts and 100 devices (Ward et al. 2002). To minimize the installation and future cost of the SAN fabric, fabric planning such as network over-subscription and choice of design topology require appropriate consideration.

1.1 Core-Edge SAN Architecture

It is common practice for SAN designs to be based on *reference* topologies. Reference topologies are adaptable structures that incorporate characteristics considered important in a network design. These topologies incorporate scalability, reliability and performance while minimizing the cost attributed to excess connection media. A number of reference topologies are available to aid in the design of SAN fabric.

Core-Edge is a reference topology considered by many to be one of the most versatile forms of SAN design. This is illustrated by its wide application throughout industry. Core-Edge SAN designs are characterized by a symmetrical structure and a high level of scalability and reliability. A contemporary Core-Edge topology is formed around a set of high bandwidth director switches (*core* switches). Each core switch connects to all switches on an additional layer of lesser bandwidth (*edge* switches), which collectively connect all hosts and devices. The effective implementation of

Core-Edge design allows no host-device pair to be greater than four intermediate connections (*hops*) away from each other. Figure 2 illustrates a generalized Core-Edge SAN topology. This SAN utilizes a separate set of edge switches for hosts and devices. This is not always necessary, but we will refer to this specific form of Core-Edge topology here.

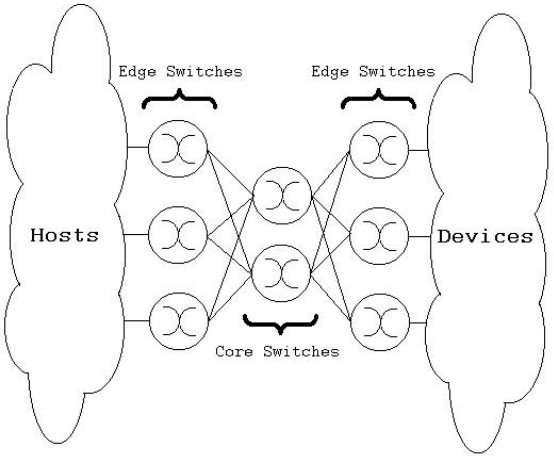

Figure 2: Generalized Core-Edge SAN design

Core-Edge SANs are easily configured to incorporate reliability and fault prevention. In Figure 2, two disjoint paths are formed between all pairs of edge switches. This is achieved through links connecting to/from these edge switches to both core switches. This can be extended with an effective connection plan to allow multiple independent paths between each host and device. If a fault was to occur within the network to disrupt a path between host and device, this connection remains active on the remaining independent paths.

The scaling of Core-Edge SANs can be achieved simply by the inclusion of additional edges switches (and core switches where required). This provides additional ports to connect host and/or devices to the network.

1.1.1 Link Over-Subscription

Link over-subscription is a practice common in network design that considers the cost benefit of merging multiple links into one. In the case of Core-Edge SANs, a link over-subscription ratio is defined as the ratio of total bandwidth entering an edge switch to total bandwidth leaving an edge switch in the direction of the core. The commercial standard for SAN links bandwidth is currently 1-2 Gb/s. In practice this bandwidth capacity is generally well beyond the instantaneous loading of a link. It is therefore considered a valid practice to employ link over-subscription within a SAN design to reduce the cost attributed to the SAN fabric. This is implemented in Core-Edge SANs using a maximum over-subscription ratio that each edge switch must not exceed. In industry this ratio is generally of order 7:1 (4:1 for high performance SANs).

1.2 Core-Edge SAN Fabric Design Problem Description

A related paper, (Ward et al. 2002), poses an integer programming (IP) formulation to a generalized SAN design problem. This problem involves finding an optimal arrangement of fabric components to simultaneously satisfy a set of bandwidth requirements between host and device pairs *flows*. In practice it is difficult to obtain *a priori* knowledge of these flows. A more viable approach investigates connecting all hosts and devices at minimum cost.

The Core-Edge SAN fabric design problem involves finding an optimal Core-Edge arrangement of fabric components to connect all available host and device ports to the SAN at minimum cost. A resulting SAN design is to satisfy all port bandwidth requirements and not exceed a maximum link over-subscription.

The Core-Edge SAN fabric design problem is described in terms of hosts, devices, switch types, and link types.

A company has a set of hosts $\mathcal{H}$ that run applications which access data from a set of devices $\mathcal{D}$. The hosts and devices have a fixed number of slots where ports are to be added. This means that a host $h \in \mathcal{H}$ or device $d \in \mathcal{D}$ may have no more than $\eta(h)$ or $\eta(d)$ ports respectively. Each port available for host h (device d) has a set bandwidth capacity $\beta(h)$ and cost $\pi(h)$ ($\beta(d)$ and $\pi(d)$, respectively). An appropriate maximum over-subscription ratio (κ) is known for the SAN.

The number of switches and links are not known *a priori*, however there exist sets of different switch types $\mathcal{S}$ and link types $\mathcal{L}$. Each switch type $t \in \mathcal{S}$ has an associated cost $\gamma(t)$ and port cost $\pi(t)$. Similarly each link type $t \in \mathcal{L}$ has cost $\gamma(t)$. We need to determine how many of each switch or link type to use and how to configure the network.

1.3 Previous Research

Commercial tools currently exist in industry that approach the problem described in Section 1.2. Hewlett-Packard (HP) laboratories currently have a heuristic tool for generating SANs with no set topology. This tool is capable of designing a SAN with 50 hosts and 100 devices, the size of the largest SANs in industry. One example from (Ward et al. 2002) shows that a heuristic tool of this nature can provide substantial savings over manual design ($1.4m versus $4m). For additional information we refer you to (Ward et al. 2002).

A limited number of tools are currently available that consider a specific topology, such as Core-Edge. One such tool is SANTK (*Storage Area Network Toolkit*) developed by the University of Missouri to aid in the design of SAN fabric. SANTK incorporates a Core-Edge SAN design tool. This tool is algorithm-based and matches feasible switch and link arrangements to user-prescribed parameters. All Core-Edge SAN topologies are generated *a priori* using a limited set of switch types and a single link type.The designs generated by SANTK are limited to the circular Core-Edge topology, where all hosts and devices are connected to one set of edge switches. The algorithm considers only the number of ports required on the edge switches and provides no connection plan for linking the hosts and devices to these switches. For

additional information we refer you to (Strand 2002).

Problems of this nature can also arise in a variety of other fields. A problem defined by a set of transshipment demands with similar flow and node restrictions can be formulated in an identical manner to the Core-Edge SAN design problem. Such problems arise in contexts such as transportation networks, telecommunication networks, distributed computer networks, distribution systems and energy systems.

1.4 Outline

The rest of this paper is organized as follows. Section 2 summarize the IP formulation for the Core-Edge SAN design problem. We present preliminary results in Section 3 and compare the IP to similar tools currently available for Core-Edge design. Finally, in Section 4 we offer conclusions regarding the validity of our approach and outline any future work that is required.

2 Core-Edge SAN Model

The Core-Edge SAN design problem is an example of a restricted network design problem, a generalization of a known NP-hard problem. This problem considers the assignment of switches and links subject to degree, node and bandwidth constraints.

2.1 Switch and Link Sets

The switches are partitioned into three sets: *host edge* switches S^H; *device edge* switches S^D; and *core* switches S^C ($S = S^H \cup S^D \cup S^C$). The *nodes* of the network are the hosts, devices and switches, i.e., $N = S \cup \mathcal{H} \cup \mathcal{D}$.

Each link $l \in L$ has a source $\sigma(l) \in N$ and a destination $\tau(l) \in N$. The links are partitioned into 4 main sets: the links from hosts to host switches; the links from host switches to core switches; the links from core switches to device switches; and the links from device switches to devices. The second and third sets are partitioned further, so that there is a defined set of links between each host and core switch, and between each core and device switch. The sets of links are: L^H between the hosts and host switches; $L^{(h,c)}$ between host switch $h \in S^H$ and core switch $c \in S^C$; $L^{(c,d)}$ between core switch $c \in S^C$ and device switch $d \in S^D$; and L^D between the devices switches and the devices. Therefore, the source and destination nodes for each set are restricted: $\sigma(l) \in \mathcal{H}, \tau(l) \in S^H$ for $l \in L^H$; $\sigma(l) = h, \tau(l) = c$ for $l \in L^{(h,c)}$; $\sigma(l) = c, \tau(l) = d$ for $l \in L^{(c,d)}$; $\sigma(l) \in S^D, \tau(l) \in \mathcal{D}$ for $l \in L^D$. Note that we have defined enough links in L^H (L^D) to connect all the ports from any host (device, respectively) to any host switch (device switch, respectively).

2.2 Decision Variables

The primary decision variables for the Core-Edge SAN model are the existence of links and switches as well as their properties (bandwidth, number of ports, etc).

Given a candidate set of links L and switches S, we need to decide which links and switches to keep in the network. Define

$$\begin{array}{ll} u_s & = 1 \text{ if we keep switch } s \in S \\ & = 0 \text{ otherwise} \end{array} \qquad \begin{array}{ll} v_l & = 1 \text{ if we keep link } l \in L \\ & = 0 \text{ otherwise} \end{array}$$

Also, define $\eta(s)$ as the number of ports on switch $s \in S$; $\beta(s)$ as the port bandwidth of ports on switch $s \in S$; $\beta(l)$ as the bandwidth of ports on link $l \in L$.

Selecting Switch and Link Types Since switches and links are selected from pre-defined types (rather than configured independently) and their cost is determined by these types, we need to select the appropriate type for each switch and link depending on their properties. This is done effectively by first creating two *technology tables*. These tables define:

1. The least cost switch for all feasible pairs of number of ports and port bandwidth
2. The least cost link for all feasible bandwidths

Using these tables, the necessary master-slave constraints, and the variables

$$\begin{array}{ll} x_{st} & = 1 \text{ if switch } s \in S \text{ is type } t \in \mathcal{S} \\ & = 0 \text{ otherwise} \end{array} \qquad \begin{array}{ll} y_{lt} & = 1 \text{ if link } l \in L \text{ is type } t \in \mathcal{L} \\ & = 0 \text{ otherwise} \end{array}$$
$$n_{st} = \text{ the number of ports of switch type } t \in \mathcal{S} \text{ on switch } s \in S$$

we can obtain the correct switch type and switch type ports for switches, and the correct link type for links. This in turn gives the correct cost for switches and links using only their properties. It is noted for $s \in S$ if $u_s = 0$ then $x_{st} = 0\ \forall t \in \mathcal{S}$. Similarly for $l \in L$ if $v_l = 0$ then $y_{lt} = 0\ \forall t \in \mathcal{L}$. The details of the constraints needed to implement the technology table are omitted here for brevity. We refer you to (O'Sullivan, Walker, and Allard 2002) for further information.

For Core-Edge designs we need the same number of links between each host and core switch, defined here as a (non-negative) integer variable m^H, and the same number of links between each core and device switch, defined here as a (non-negative) integer variable m^D. We also need to ensure that we use the same link type between every host and core switch, and between every core and device switch, the variables

$$\begin{array}{ll} z_t^H & = 1 \text{ if we use link type } t \text{ between the host and core switches}, t \in \mathcal{S} \\ & = 0 \text{ otherwise} \\ z_t^D & = 1 \text{ if we use link type } t \text{ between the core and device switches}, t \in \mathcal{S} \\ & = 0 \text{ otherwise} \end{array}$$

2.3 Objective Function

The properties of the hosts and devices are known *a priori*. The properties of the switches and links determine their position on the technical table (and in the case of

switches the number of ports of any switch type). The total cost of the SAN fabric is

$$Z = \sum_{s\in S, t\in\mathcal{S}} \underbrace{(\gamma(t)x_{st} + \pi(t)n_{st})}_{\text{switch and port cost}} + \sum_{l\in L, t\in\mathcal{L}} \underbrace{(\gamma(t)y_{lt})}_{\text{link cost}} + \sum_{l\in L^H} \underbrace{\pi(\sigma(l))v_l}_{\text{host cost}} + \sum_{l\in L^D} \underbrace{\pi(\tau(l))v_l}_{\text{device cost}}$$

2.4 Node Constraints

The number of links leaving a host (entering a device) must be equal to the number of ports on that host(device, respectively). This means all host and device ports are connected by the SAN.

$$\sum_{l\in L^H, \sigma(l)=h} v_l = \eta(h), h \in \mathcal{H} \tag{1}$$

$$\sum_{l\in L^D, \tau(l)=d} v_l = \eta(d), d \in \mathcal{D} \tag{2}$$

The links leaving a host (entering a device) must have more bandwidth than the ports on the host (device, respectively). This ensures that the ports can be used to their maximum bandwidth capacity.

$$\beta(l) \geq \beta(\sigma(l)), l \in L^H \tag{3}$$

$$\beta(l) \geq \beta(\tau(l)), l \in L^D \tag{4}$$

2.5 Switch Constraints

There are constraints for ordering switches within their sets, i.e., we will use the first switch in S^H before the second, third, etc switches in S^H. There are also constraint to remove antisymmetry amongst switches in the same set. These constraints require the earlier switches to have either more ports or the same ports and more port bandwidth than later switches.

The number of links attached to a switch must be less than the number of ports on the switch.

$$\sum_{l\in L, \sigma(l)=s \text{ or } \tau(l)=s} v_l \leq \eta(s), s \in S \tag{5}$$

The port bandwidth of an edge switch must be more than the bandwidth of the links coming into the switch.

$$\beta(s) \geq \beta(l), s \in S^H, l \in L^H, \tau(l) = s \tag{6}$$

$$\beta(s) \geq \beta(l), s \in S^D, l \in L^D, \sigma(l) = s \tag{7}$$

The port bandwidth of a core switch must be more than the bandwidth of all links coming into the switch.

$$\beta(s) \geq \beta(l), s \in S^C, l \in \left(\cup_{h\in S^H} L^{(h,s)}\right) \cup \left(\cup_{s\in S^D} L^{(s,d)}\right) \tag{8}$$

2.6 Architecture Constraints

There are constraints for link antisymmetry and ordering of links within their sets, similar to those defined for switches. These constraints require that earlier links have more bandwidth than later links.

The following constraints combine with the switch ordering constraints to ensure that the switch types remain constant over each of the switch sets. Note that $s+1$ means the next switch in the set.

$$x_{s+1t} \leq x_{st}, s \in S^H, \text{ except the last host switch} \tag{9}$$
$$x_{s+1t} \leq x_{st}, s \in S^D, \text{ except the last device switch} \tag{10}$$
$$x_{s+1t} \leq x_{st}, s \in S^C, \text{ except the last core switch} \tag{11}$$

The links within a link set have the same type or are not active. Note that $l+1$ means the next link in the set.

$$y_{l+1t} \leq y_{lt}, l \in L^{(h,c)}, \text{ except the last link in each set} \tag{12}$$
$$y_{l+1t} \leq y_{lt}, l \in L^{(c,d)}, \text{ except the last link in each set} \tag{13}$$

The following constraints combine with the link type ordering constraints (12) & (13) to ensure than the link types remain constant between active edge and core switches.

$$y_{1t} \leq f_t + (2 - (u_h + u_c)), h \in S^H, c \in S^C, t \in \mathcal{S} \tag{14}$$
$$y_{1t} \geq f_t - (2 - (u_h + u_c)), h \in S^H, c \in S^C, t \in \mathcal{S} \tag{15}$$
$$y_{1t} \leq g_t + (2 - (u_c + u_d)), c \in S^C, d \in S^D, t \in \mathcal{S} \tag{16}$$
$$y_{1t} \geq g_t - (2 - (u_c + u_d)), c \in S^C, d \in S^D, t \in \mathcal{S} \tag{17}$$

The following constraints combine with the link ordering constraints to ensure the number of links between active edge and core switches remain constant.

$$\sum_{l \in L^{(h,c)}} v_l \leq m^H - M(2 - (u_h + u_c)), h \in S^H, c \in S^C \tag{18}$$
$$\sum_{l \in L^{(h,c)}} v_l \geq m^H + M(2 - (u_h + u_c)), h \in S^H, c \in S^C \tag{19}$$
$$\sum_{l \in L^{(c,d)}} v_l \leq m^H - M(2 - (u_c + u_d)), c \in S^C, d \in S^D \tag{20}$$
$$\sum_{l \in L^{(c,d)}} v_l \geq m^H + M(2 - (u_c + u_d)), c \in S^C, d \in S^D \tag{21}$$

3 Results

The IP introduced in this paper yields some clear advantages over the other tools available for Core-Edge SAN design. The most comprehensive tool currently available in this field is SANTK, described in Section 1.3. The resultant SANs design

from the IP are optimal and including a full connection plan from all host to devices. These design properties are not available in current versions of SANTK. All functionality of the Core-Edge design tool in SANTK is encompassed within the IP.

As a preliminary test for the Core-Edge IP, we apply the IP to a number of example problems of various magnitude. These example problems are presented in Table 2. The fabric sets available in each example includes 3 switch types and 2 link types as listed in Table 1. The hosts and devices are uniform, each including 2 available ports each with maximum bandwidth of 1 Gb. We apply a constant over-subscription ratio of 1:5 throughout.

Switch	1	2	3	Link	1	2
Port Number	8	16	32	Bandwidth (b)	1e8	5e7
Port Bandwidth (b)	1e8	1e8	1e8			

Table 1: Example Problem Components

The IP is formulated in AMPL and solved using CPLEX 8.1 on a dual processor AMD Athlon MP 2200+ (1 Gb RAM) using Red Hat Linux 8.0. Figure 3 summaries the performance statistics for these example problems.

Example	Hosts	Devices	Variables	Constraints
1	5	5	10092	10550
2	10	10	10392	10860
3	20	20	10992	11480
4	20	50	12192	12710
5	20	100	14192	14760
6	50	100	14792	15390

Table 2: IP Problem Statistics

Example	MIP Iterations	B & B Nodes	Solve Time (s)
1	250	0	41
2	4478	84	186
3	19755	504	294
4	12345	198	222
5	33052	277	583
6	65879	1391	1019

Table 3: IP Solution Statistics

The scale of the examples considered in Table 2 have been selected to blanket the range of SANs currently in industry. The IP is observed to perform well for all cases. The solve time required to generate each SAN design is satisfactorily small and can be considered acceptable for a commercial tool of this nature. Example 6 is of equivalent magnitude to the largest SANs currently in operation in industry. This example solves in 1019 seconds, well within allowable bounds for a SAN design of this size. A less complex example of a Core-Edge SAN generated by the IP is illustrate in Figure 3. This design corresponds to example 2 described above in Table 2. This example solves in 186 seconds.

Further work is still required to test the IP on more diverse problems, incorporating additional fabric sets and less uniform host and device port arrangements. This is currently underway, however results are not available at this time. Also it would be useful to test the IP formulation on real world designs already implemented in industry. This would provide knowledge of cost effectiveness of the Core-Edge design and the potential of this tool.

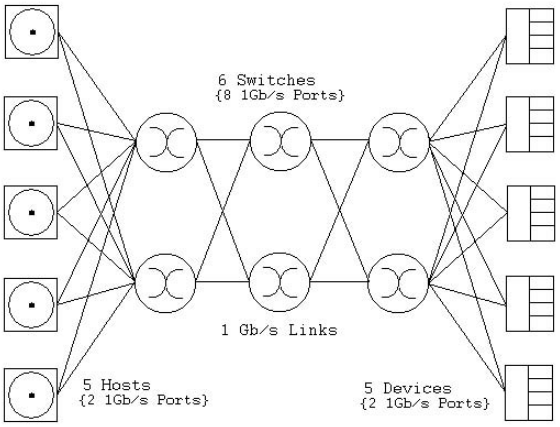

Figure 3: Generalized Core-Edge SAN design

4 Conclusions

In this paper we introduce an IP formulation to a restricted network design problem considering Core-Edge SAN design. The IP approach yields some advantages over currently available tools in this area. The IP provides the user with an optimal SAN design with a full connection plan not currently available from Core-Edge design software.

The IP was applied to a selection of example problems encompassing the scale of SANs in commercial operation. The IP was found to perform well, producing solutions for all cases in an acceptable time-frame for a commercial tool. Further testing of the IP is, however, still required for more diverse networks. Also it would be advantageous to test the IP against implemented designs in industry to analyze the cost effectiveness of Core-Edge design.

References

O'Sullivan, M., C. Walker, and J. Allard. 2002, November. "Improving Optimal Storage Area Network Design." *ORSNZ Conference Twenty Naught Two.* 119–128.

Strand, S. 2002. "Storage Area Networks and SANTK." Master's thesis, University of Minnesota Fibre Channel.

Ward, J., M. O'Sullivan, T. Shahoumian, and J. Wilke. 2002, January. "Appia: Automatic Storage Area Network Design." *Conference on File and Storage Technology (FAST'02).* 203–217.