This article was processed using the \LaTeX\ macro package with LLNCS style.


As shown in [9], this scheme yields a $VPL^n$ with load

$$\mathcal{L}(VPL) = O( h \cdot k \cdot \log N \cdot N^{-\frac{1}{k+1}} )$$

for any $k > 1$, and its stretch factor satisfies $\psi \leq 8k$.

5 Summary and extensions

In this paper we have considered a new problem of designing the layout of virtual paths in a given network to be used for VC routing. We have defined the characteristics that are important for such a layout, and a precise model for it. We have presented an optimal construction scheme for fairly simple networks first, and extended it to more general networks. We have also shown a sub-optimal scheme, which may be applied to any network.

More work is needed in finding better schemes for general networks, and in extending the existing schemes for supporting fault tolerance in the network. The characteristics of a good layout may be extended to the case when some statistics regarding the volume of connections between every pair of switches is known.

References

parameter of the algorithm).

- The distance between every pair of vertices in each cluster is not more than \((2k - 1) \cdot m\).

- Each vertex is included in not too many clusters (i.e. \(O(k N^{1/k})\)).

In each cluster we choose its center as a pivot (i.e. the vertex which is closest to all the other vertices of the cluster), and connect all the other vertices of the cluster to the pivot by a \(VPL^{1-n}\) with \(\psi = 1\) and \(h' = \frac{n}{k}\). Clearly, every pair of vertices whose distance is less than \(m\), belong to at least one common cluster, and may be connected using no more than \(h\) hops via the pivot of that cluster.

We repeat this scheme for increasing parameter \(m\) (until the algorithm yields one cluster which includes the whole network). In order to minimize the stretch factor \(\psi\), each pair is connected using the \(VPL^{1-n}\) of the smallest common cluster. Refer to Figure 4 for a graphic demonstration of this scheme.

![Figure 4. VPL^{1-n} on a general graph](image-url)
Note that two switches that reside in different subtrees may be connected using no more than \( h \) VPs, by going from one switch to the pivot (no more than \( \frac{h}{2} \) hops), and from the pivot to the other switch (again, no more than \( \frac{h}{2} \) hops); and that this route is the shortest possible (hence \( \psi = 1 \)). If the switches reside in the same subtree, then they are catered by the recursive application of the scheme in that subtree.

The load on each edge is composed of the load of the \( \text{VPL}^{1-n} \) to the pivot and the load of the recursive application of \( \text{VPL}^{n-n} \) in the subtree. Since an edge participates in one such \( \text{VPL}^{1-n} \) and one smaller \( \text{VPL}^{n-n} \) (of its subtree), its load is bounded by the by the following recurrence equation.

\[
\mathcal{L}(\text{VPL}^{n-n}(h, N)) \leq \mathcal{L}(\text{VPL}^{n-n}(h, N/2)) + \mathcal{L}(\text{VPL}^{1-n}(h/2, N/2))
\]

This equation yields the solution \( \mathcal{L}(\text{VPL}) \leq 4hN^{1/2} \) as may be easily verified. Again, this result is asymptotically optimal as proven in [9].

4.2 General networks

We cannot use the previous scheme for general networks, since there does not necessarily exist a set of switches, through which all the rest are connected. The new construction scheme is based on a technique of [1], for the construction of "regional routing schemes" (used for regular routing problems). The scheme involves many new concepts, and we present only an intuitive outline of it.

A central concept here is the \( M \)-neighborhood of a vertex \( v \): This neighborhood is a set of vertices \( w \) with \( d(v, w) \leq M \). The scheme is based on a clustering algorithm that receives two integers \( k, m \) as parameters, and divides the vertices of a graph into overlapping clusters, with three important properties:

- For each vertex \( v \) there is a cluster that includes \( v \), and all its \( m \)-neighborhood (recall that \( m \) is a
4.1 Tree networks

We need the following theorem (from [9]) for a VPL\(^{1-n}\) for trees.

**Theorem 5.** Let \(G\) be a tree network with \(N\) switches, \(\psi = 1\), \(h > 1\). Then there exists a VPL\(^{1-n}\) with \(\mathcal{L}(VPL) \leq 2hN^{\frac{\psi}{\psi}}\).

We also need the following graph-theoretical result (from [14]) for choosing the pivot.

**Theorem 6.** Let \(G\) be a tree with \(N\) vertices. There exists at least one vertex \(v\) (called a median), whose removal divides the tree into subtrees which are not greater than \(\frac{N}{2}\).

The scheme is as follows: First choose a median \(m\) of \(G\) as a pivot. Construct a VPL\(^{1-n}\) with \(h' = \frac{1}{2}h\) in each subtree, with \(m\) as its root. Build a VPL\(^{n-n}\) with \(h' = h\) in each subtree. The reader is referred to Figure 3.

![Diagram showing the construction process for VPL networks on a tree.](image-url)
Remark. From the above discussion, it’s easy to see that if we allow to exhaust the VP routing tables in the port-adaptors, we can construct networks of size $N = 1362$ with only two hops, or enormous $N = 2^{13}$ networks with three hops. Similar results hold even if the VP table is per switch.

3.3 The Mesh network

Recall that an $x \times y$ mesh network of size $xy$ is comprised of horizontal chains of size $x$ and vertical chains of size $y$. Similarly, an $x \times y$ torus network is comprised of rings of size $x$ and $y$ respectively. We build a $VPL^{x\times y}$ for these networks by building a $VPL^{x\times y}$ for each horizontal and vertical component and taking the union of all these layouts. Note that the edges of different components are distinct, and hence the load on an edge is determined by the layout of the component alone.

To achieve a low $\mathcal{L}(VPL)$, one has to take $x$ and $y$ into account, when determining the hop counts $h_x, h_y$ for the horizontal/vertical components. The following choice for $h_x, h_y$ is optimal:

$$h_x = \frac{h}{\log x} + 1; \ h_y = h - h_x$$

The hop count is restricted by $h$, since any switch can be reached by no more than $h_x$ hops (to get to the correct vertical position) and no more than $h_y$ hops in the vertical $VPL$. The load $\mathcal{L}(VPL)$ is $\mathcal{L}(VPL) = x^{\frac{h}{\log x}} = y^{\frac{h}{\log y}}$. Note that for the simple case where $x = y = \sqrt{N}$ (i.e. a "square"), $\mathcal{L}(VPL) = hN^{\frac{2}{h}}$ - as for ring networks.

4 More Complex Networks

In this section we extend the construction scheme of the previous section for more complex networks, namely tree shaped networks. In addition, we outline a different technique for constructing a $VPL$ on general networks; This technique is sub-optimal in the usage of the communication network (i.e. $\psi > 1$).
and the pivots are connected to each other by $VPL^{n-n}$ with $h = 1$ (meaning a VP between every pair of pivots). The scheme is demonstrated in Figure 2.

The load of the construction satisfies the following recurrence formula

$$\mathcal{L}(VPL) = \mathcal{L}(VPL^{n-n}(h, N)) \leq \mathcal{L}(VPL^{n-n}(h, N^{1-\frac{1}{\hat{h}}})) +$$

$$2\mathcal{L}(VPL^{1-n}(\frac{h-1}{2}, N^{1-\frac{1}{\hat{h}}})) +$$

$$\mathcal{L}(VPL^{n-n}(1, N^{1-\hat{h}}))$$

and therefore $\mathcal{L}(VPL) \leq \frac{h(h+1)}{2} N^{1-\hat{h}}$ (the exact considerations for this construction, as well as the formal calculations may be found in [9]). In [9] we have proven that this construction is asymptotically optimal.

Fig. 2. $VPL^{n-n}$ on a ring
1. A $VPL^{1-n}$ that connects each pair of switches that are both part of the same sub-network (this is typically a recursive application of the whole construction scheme).

2. Each pivot $p$ is a root of a $VPL^{1-n}$ in every sub-network that is adjacent to that pivot. This $VPL^{1-n}$ connects all the switches in this sub-network to $p$.

3. The pivots are connected in a $VPL^{n-n}$ network, which is used in combination with the previous network to connect switches from different sub-networks.

In the case of a ring - we divide the $N$ size ring into $N^{1/h}$ chains of size $N^{1-1/h}$ each. The switches which are part of each chain are interconnected by applying a similar $VPL^{n-n}$ scheme with $h$ hops on a smaller network. Each pivot is connected to the two chains adjacent to it by $VPL^{1-n}$ with $h' = \frac{h-1}{2}$.

---

In this case, it is not the same recursive construction, since each segment is a chain and not a ring, but it is almost identical, except for the two ends of the chain, which are not connected in two directions.
3.1 Constructing a $VPL^{1-n}$

Let $G$ be a ring of $N$ switches, with a root $r$. Due to remark 3, the ring may be seen as two separate chains of switches, each of size $\frac{N-1}{2}$, since each switch may access the root through the shortest route only\(^7\). Therefore it suffices to concentrate on a chain of $N$ switches with the root at one of the ends of the chain.

Let $h = 2$, construct $VPL^{1-n}$ in the following way (see Figure 1): First divide the chain into $\sqrt{N}$ equal sections of size $\sqrt{N}$. Call the switch that is closest to $r$ in section $i$ the "pivot of section $i".

Connect $r$ to all pivots by VPs; Connect each pivot $i$ to all the switches in section $i$;

It is clear that $r$ can be reached from every other switch by using two VPs (hence $h = 2$): one VP to the pivot of the section, and one VP from the pivot to $r$. It is also clear that this route is shortest in terms of the underlying chain (hence $\psi = 1$).

The load of the layout satisfies $L(VPL) \leq 2\sqrt{N}$, since at most $\sqrt{N}$ VP pass through an edge to the pivot of the current section, and at most $\sqrt{N}$ VPs, from the pivots to $r$.

As demonstrated in Figure 1, the results can be extended to $VPL^{1-n}$ with any $h$, by dividing the chain into sections of size $N^{1-1/h}$, connecting the $N^{1/h}$ pivots to the root, and connecting each pivot to its section by a $VPL^{1-n}$ with $h' = h - 1$ (a recursive application of the scheme). The load of this scheme is $L(VPL) \leq hN^{1/h}$ (See [6] for a formal description and analysis).

3.2 Constructing a $VPL^{n-n}$

Our construction scheme is based on dividing the network into smaller sub-networks (as for $VPL^{1-n}$), which are interconnected through a small number of switches (called pivots). The full layout is comprised out of three separate layouts:

\(^7\) If $N$ is odd then the switch which is furthest of $r$ has two alternatives, but we ignore this detail since it does not change the discussion substantially.
access pattern, namely when all switches connect to a single chosen switch (called the root); We term this access pattern "1–n access". The VPL for n–n access will be denoted by VPL^{n-n}, while a VPL for 1–n access will be denoted by VPL^{1-n}. Definition 2 and 3 are redefined in a straightforward way for VPL^{1-n}.

Remark. Finding a VPL^{1-n} is easier than finding a VPL^{n-n} as implied by the following facts:

1. Every feasible VPL^{n-n} is also a feasible VPL^{1-n}, but the reverse is not true.
2. The load L(VPL^{n-n}) of an optimal VPL^{n-n} is never less than the load of an optimal VPL^{1-n}.

Besides its methodical value as an easier problem to be tackled first, VPL^{1-n} has practical importance, as it may prove useful for server networks, where data flows from a center to different destinations and vice versa. An example for this is a video conferencing server - which has connections to all users who are currently engaged in a video conference [11]; Another example is an interactive TV station - which is engaged in many separate sessions with different users.

3 Simple networks: Rings and Meshes

We first discuss ring networks, and then use the construction to build a VPL^{n-n} for a mesh network. This construction exemplifies a general technique for constructing a VPL^{n-n} for networks that can be decomposed into simpler networks.

Remark. In this section and in the next one we restrict ourselves to ψ = 1. This stretch factor does not allow any inefficiency in the usage of the underlying communication network - namely for every pair of switches there must exist a path in GVP with less than h VPs, which is also a shortest path in G.

---

6 This is not to be confused with a multicast service, where all destinations receive the same data from a given source, while here we discuss separate streams of data from a service center.
Underlying this definition is the assumption that a VP routing table exists in every port adaptor, following the switch architecture of [4]. Also we assume that each VP is a bidirectional route, comprised of two unidirectional routes in opposite directions, (this assumption is accepted in the literature as it substantially simplifies the network management). The rest of the discussion applies, with minor changes, in the case when there exists a single table in the switch, or when the tables in the port adaptors of the switch are identical (see [5]).

**Definition 2.** Let \( \psi \geq 1 \) be a real number called the *stretch factor*. Define the *hop count* \( H(p) \) for a simple path \( p \) in \( G_{VP} \), as the number of VPs that form the path.

Extend the definition to a pair of vertices by defining \( H_\psi(v, w) \) as the minimum hop count \( H(p) \) for all paths \( p \) connecting \( v \) and \( w \) in \( G_{VP} \), whose induced path \( \Gamma(p) \) is shorter than \( \psi \cdot d_G(v, w) \) (\( d_G(v, w) \) being the shortest distance between \( v \) and \( w \) in \( G \)).

Finally define \( H_\psi(G) = \max_{v, w \in V(G)} H_\psi(v, w) \).

Note that \( H_\psi(v, w) \) is the minimum number of VPs that may be used to form a VC between \( v \) and \( w \), such that the length of that VC will not be too large with respect to the minimal route between \( v \) and \( w \) (the meaning of "too large" depends on the stretch factor).

**Definition 3.** A VPL is *feasible* with respect to \( (\psi, h, G) \) iff \( G_{VP} \) is connected and \( H_\psi(G) \leq h \).

A feasible VPL will be denoted by \( VPL(\psi, h, G) \), or by \( VPL(h, N) \) when \( G \) and \( \psi \) are understood from the context and \( |V| = N \).

**Definition 4.** A VPL is *optimal* if it is feasible and its load \( L(VPL) \) is minimal among all other feasible VPLs.

So far we have discussed the case when it is required to connect every switch to every other switch; We term this access pattern "n–n access". We have found useful it to focus first on a more restricted
this facility must be used only in a small number of switches along the route, rather than in every intermediate switch, substantially speeds up the process [4, 13].

**Fault Tolerance:** The VP layout must keep the above properties even in face of hardware failures in switches and in communication lines.

**Restricted load:** As mentioned above, the VP routing tables are limited in size, and the layout must utilize them in a way that not too many VPs go through a switch.

We now present an exact mathematical model and formal definitions of the above properties, based on graph theory (see [6] for the basic terminology). For the sake of simplicity we shall not present here the precise formal definitions, but a more intuitive version. Exact definitions as well as a detailed discussion, may be found in [9].

In our model we have an underlying communication network, which consists of switches and links between them. This network is modelled by a simple undirected graph $G = (V, E)$.

Let $\mathcal{P}_G$ be the set of all simple paths in $G$. The virtual path layout $VPL$ is represented by a graph $G_{VP} = (V, E_{VP})$ and a function $I : E_{VP} \rightarrow \mathcal{P}_G$ where the vertices of $G_{VP}$ are the same as of $G$, and each edge $e \in E_{VP}$ represents a VP between two switches. Each VP is mapped to a simple path in $G$ by $I$. We term this path the *induced path* of the VP.

**Definition 1.** The *load* $L(e)$ on a link $e \in E$ is equal to the number of VPs $x \in E_{VP}$ that include $e$ in their induced paths (namely $e \in I(x)$). The *load of a layout*, $L(VPL)$ is the maximum load $L(e)$ over all $e \in E$.

---

4 It is important to have low setup times if the connection is used for a relatively short time, to accommodate for an ATM version of datagrams; Another reason is that the lower the setup time is, the smaller are the chances that two setup procedures complete simultaneously for the same resource in the network (an event that may cause deficiencies during bandwidth allocation [5]).

5 This property is out of the scope of this paper, and we present it here for the completeness of the list of desired properties of a VP layout.
to a more complex family of networks, namely trees. In Section 5 we outline a different sub-optimal technique for general networks (based on advanced graph theoretical results [1]). Finally, we summarize the results and suggest further research directions in Section 6.

2 Problem definition

Before devising schemes for designing a VP layout, one has to define precisely the characteristics that make such a layout good, in terms of its performance. We argue that the following properties are important in such networks:

**Virtual channel structure:** A virtual channel is composed of a concatenation of VPs. This assumption includes the case when there may exist a VC which is not part of any VP (simply add a VP for every physical link), and is more general than the case when a VC may use a single VP - take the VP count (defined below) to be one.

**Full switching capability:** Each switch can switch both VPs and VCs (the homogeneous model mentioned above). This assumption simplifies the network structure, while not substantially complicating the switch itself.

**Full reachability:** Each pair of switches may be connected by a VC; This assumption allows to connect every pair of terminals, regardless of the switch that connects them to the network.

**Short underlying route:** The chosen route for a VC must be short in terms of the underlying communication network, to efficiently utilize it.

**Restricted VP count:** The number of VPs that are used by any VC should be relatively small. This fact helps in reducing the connection setup time and the switch processing time, since the VC routing tables should be updated only when a VP ends (and the VC must be routed into another VP). Since the update of these tables is done by a network layer facility - in software - the fact that
as long as the new VPI is non-empty. Only when the VPI is empty, is the VCI considered, to allow the separation of the different connections. The VC table determines the new VCI label and the output port (as in the VP table), but also a new VPI into which the VC is multiplexed.

In this paper we have focused on the homogeneous model for ATM switching, since it is more flexible, and does not seem to substantially complicate the switch architecture. A broader discussion on B-ISDN and ATM as well as a discussion on the heterogeneous model routing mechanism can be found in [2, 10]. A detailed description of the homogeneous model routing mechanism in ATM can be found in [4].

The VC and VP concepts have been extensively discussed in the literature, however, to the best of our knowledge, the problem of how should VPs be laid out in a given communication network was never addressed. While the simple solution for this problem, of having a VP between every pair of switches is ideal for fairly small networks \(^3\), it is not suitable for bigger networks, which are likely to be constructed in the future. For the same reason, we also assume that a VC may extend over multiple VPs (rather than a single VP).

This work focuses on techniques for designing the layout of VPs in a given communication network. To this end, we first define (in Section 2) our assumptions and the essential characteristics of a good layout. This layout may serve as an initial setup of VPs in a network, and may still be extended dynamically, according to the future needs of the network, while keeping the "good" performance of the initial setup. We also define and solve an easier problem - that helps in solving our layout problem.

We then examine (in Section 3) several simple (yet practical) networks, namely rings and grids (or meshes), show how to construct a layout of VPs for them. The techniques are extended in Section 4

\(^3\) In certain network topologies, some of the VP routing tables may be filled with \(\frac{1}{2}N(N-1)\) entries, in a network of \(N\) switches, and since the VPI is restricted to 12 bits, \(\frac{1}{2}N(N-1) \leq 2^{12}\) or \(N \leq 91\) - if we allow to fill the VP table up to its total capacity, which is not likely to be the case (see [9] for a comprehensive discussion); Furthermore, switching elements are not restricted to full capacity switches, and may well be simple (and abundant) devices as multiplexors.
ATM is based on small fixed size packets, which are called cells. Because of the very high requirements for switching rates, the routing of the cells must be done by a dedicated hardware, implying very simple routing algorithms. The routing scheme chosen in ATM is based on two fixed length labels in the header of each cell (VCI and VPI). These labels serve as indices into the routing tables of the switches, and they determine the route that a cell will take in the network. For this reason, two cells that arrive at a switch with equal VCI and VPI labels will take the same route in the network, and will effectively become part of the same connection; This connection is termed virtual channel (or VC). Conversely, two cells that arrive at the same switch but belong to separate connections between end-users should differ in at least one of these labels.

The concept of the virtual path (VP) is a later concept, whose main role is to avoid overflowing the VC tables of the network switches, and to simplify various network management aspects. This path multiplexes many VCs that share the same route, requiring a single VP entry in every intermediate switch, instead of a separate VC entry in the original scheme. This is achieved by a layered routing scheme, in which many switches along the route ignore the VCI of cells and perform the switching according to the VPI alone, while only a small number of switches considers the VCI. This VP/VC hierarchy may be implemented by two different models:

The heterogeneous model: The VP switches are separate entities and do not perform VC switching at all. The VCI (sometimes combined with the VPI as well) is considered only when the cell arrives at a VC switch.

The homogeneous model: Each switch can switch both VPs and VCs. This may be implemented by an architecture in which VP and VC tables reside in every switch in the network [4]. When a cell arrives at a switch, its VPI is used to determine the next VPI (that will replace the current one), and the output port into which it is switched; During this process the cells’ VCI is ignored
Designing the Virtual Path Layout in ATM Networks *

I. Cidon¹, O. Gerstel² and S. Zaks²

¹ IBM T.J. Watson Research Center
Yorktown Heights, NY 10598
E-mail: cidon@watson.ibm.com

² Computer Science Dept.
Technion, Haifa 32000, Israel
E-mail: orig,zaks@cs.technion.ac.il

Abstract. We develop methods for designing the layout of virtual paths on a given ATM network. We define precise measures for determining if a given layout uses the network efficiently, does not over-utilize the routing tables, and provides short setup time for every connection request. We then present a scheme for designing the layout of virtual paths for relatively simple networks (rings, meshes) and extend the results to a wider class of networks (general trees). We conclude by presenting a different technique, which is not as efficient but may be applied to a general network.

1 Introduction

The Asynchronous Transfer Mode (ATM) [10, 2] is the transmission, switching, and multiplexing technique chosen by CCITT for B-ISDN. Due to the future importance of fast, broadband, integrated networks, ATM has been extensively discussed in recent years.

* Submitted to INFOCOM’94