Dynamic Wavelength Allocation in All-Optical Ring Networks*

Ori Gerstel (corresponding author)
IBM T. J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, NY 10532,
ori@watson.ibm.com, Phone: (914)784-7193, Fax: (914)784-6225

and

Shay Kutten
Industrial Eng. Dept., Technion, Haifa 32000, ISRAEL,
kutten@ie.technion.ac.il, Phone: (+972)4829-4505, Fax: (+972)4823-5194

Abstract

We focus on wavelength allocation schemes for all-optical ring networks deploying wavelength division multiplexing without wavelength conversion capabilities. We restrict ourselves to worst case performance analysis of the network, an approach which yields robust performance guarantees. For an N node network, we first consider a "static" wavelength allocation scenario, in which all required lightpaths (connections) are known in advance. We prove that if the maximum number of lightpath requests which use any link in the ring (termed the "load") is bounded by $L_{\max}$, then up to $2L_{\max} - 1$ wavelengths may be needed. This result matches the known upper bound which shows that $2L_{\max} - 1$ wavelengths are sufficient. Next, we consider a "dynamic" scenario, in which requests to add or delete a lightpath arrive at different times, and the arrival/departure process is not known. We prove that the shortest path routing heuristic produces a routing which has at most twice the load $L_{\max}$ of the optimal solution. As far as the wavelength allocation problem is concerned, we show that at least $0.5L_{\max} \log_2 N$ wavelengths are required, and develop an algorithm which requires $2L_{\max} \log_2 N$ wavelengths in the worst case. For the case when the load is high and blocking is necessary we present two improved algorithms, both of which result in delaying the first blocking event, and in less blocking overall. Our results show that dynamic scenarios result in substantial degradation in the utilization of wavelengths comparing to static or even semi-dynamic scenarios.

Keywords: Optical networks, wavelength allocation, routing, worst case analysis

*This work was supported in part by grant MDA 972-95-C-0001 from ARPA. A preliminary version of this paper was published as an IBM research report no. RC20462, May 1996

1 Introduction

1.1 Background

As WDM systems start emerging from laboratories and being deployed in commercial contexts, the main resource allocation problem associated with WDM networks — of efficiently allocating wavelengths to lightpaths — becomes of crucial importance. The focus of most past research on the allocation of wavelengths in WDM networks (e.g, [1, 2, 3, 4] and several works in [5]) has been based on a combination of the following approaches: (1) maximalistic approaches, in which optimization of multiple parameters of this hard problem is attempted, (2) heuristic approaches, in which the performance of the solution is demonstrated, but no guarantees are proven, (3) probabilistic approaches, in which the arrival/departure process of lightpaths is known (or assumed), (4) provably suboptimal approaches, in which wavelength allocation is far from optimal, and (5) higher network level view, in which the lightpaths are assumed to be required for the use of a single higher level network (e.g., ATM), and thus the needs of that network are optimized together with the lightpath problem.

The main drawback of most of the above approaches stems from the severe limitations that current optical technologies impose on the amount of available wavelengths per fiber. While experimental systems report large number of up to 100 wavelengths per fiber [6], current state-of-the-art manufacturing processes restrict the number of wavelengths per fiber of commercial WDM multiplexers to as low as 4 (Pirelli), 8 (Lucent Technologies), and up to 20 (IBM). On the other hand, the lack of commercial optical wavelength converters currently overrules better reuse of wavelengths. As a result, this very limited resource must be exploited most carefully to achieve low blocking probabilities of lightpath requests.

While approach (5) attempts to alleviate this bottleneck by designing the higher level network to optimize the number of wavelengths, it is not suitable in situations in which the lightpath provider is not aware of or is not involved in the management of higher level networks.

The current research takes a different approach. Instead of focusing on general topologies but restricted arrival/departure processes of lightpath requests, we assume no knowledge on these processes

(nor any other considerations that restrict the request pattern) but restrict the topology to rings only. We believe that this topological restriction still yields results of high practical value since ring networks are the predominant topology for current MAN/Interoffice networks, and are thus expected to be the first topology to be used for WDM networks outside laboratories and testbeds. We also focus on worst case analysis of the problem as it is the only approach which guarantees robust bounds on the performance of the system. This restricted model enables us to study in greater depth the behavior of the problem and achieve almost tight bounds for different scenarios.

We separate the *routing* problem, of determining which part of the ring should be used to connect the source and destination of a lightpath request, from the *wavelength allocation* problem, of assigning a wavelength to each route. This technique is justified for the following reasons:

- The network users may choose to have control on the routing to support fault tolerance (namely, two routes may require disjoint paths as they are responsible for backing up each other),
- Additional considerations, such as constraints on propagation delays may require some route to take the short alternative around the ring,
- There exists an optimal algorithm for solving the static routing problem along on a ring [7], namely, the case in which all the requests are known in advance. For the dynamic case (in which case the requests arrive at different times and the algorithm has to react without knowledge of the future) we prove that an almost optimal heuristic is to use shortest path routing. These algorithms may be employed as a first stage, and used as an input to the wavelength allocation problem,
- Computationally efficient solutions to the combined routing and wavelength allocation problem which allocate resources optimally are not plausible [8].

1.2 Related works and contribution of the paper

Substantial analytical research has been carried out in recent years on routing and wavelength allocation problems for lightwave networks, starting with classic graph-theoretical problems of coloring the nodes

of *interval graphs* and *circular arc graphs*, which correspond to wavelength allocation in chain and ring networks respectively, through various models of tree topologies [2, 9, 10], and general mesh topologies [3, 9]. On the other hand, numerous heuristic techniques have been proposed for the design of lightpaths in such systems [11, 12, 13, 14, 1, 15, 16].

Even for ring networks the composite problem of finding a route for each lightpath request and allocating a wavelength to it is very hard (proven to be NP-complete in [8]). It is therefore reasonable to split the problem into two phases: (1) finding a route for each lightpath request and (2) allocating a wavelength to it.

As far as the first problem is concerned, an optimal algorithm exists for the static design problem [7]. The dynamic case for this problem is considered in the present work, in which we show that the simple shortest-path heuristics is up to twice away from the optimal solution.

As far as the second problem is concerned, given a sequence of lightpath requests and the physical route for each of them, let the maximum *load* on any link in the network (denoted $L_{\max}$) be the maximum number of these routes that share a link. This characteristic of a lightpath request system turns out to be the major factor in the worst case analysis of the system. Clearly, $L_{\max}$ is a lower bound on the number of required wavelengths since each lightpath that shares a link with maximum load must be assigned a different wavelength.

Focusing on ring networks, the best known approximation for the static case (as far as worst-case bounds are concerned) requires up to $2L_{\max} - 1$ wavelengths in the worst case [17]. This upper bound is matched by a worst-case scenario presented herein which indeed requires $2L_{\max} - 1$ wavelengths.

In many cases, it is unreasonable to assume that all lightpaths are given in advance, and more dynamic scenarios must be considered. A first step in this direction is the *semi-dynamic* scenario, in which lightpath addition requests arrive into the system with some unknown distribution, however no lightpaths are deleted. For this case, [18] has presented an algorithm which requires $W \leq 3L_{\max}$ wavelengths. A matching lower bound of $W \geq 3L_{\max} - 2$ is presented in [19].

However, for practical purposes it is unreasonable to assume that lightpaths are not deleted from the

network. Thus, it is necessary to study *fully-dynamic* scenarios. For this case the only known results are based on probabilistic models, assuming standard Poisson arrival processes [14]. In the current paper we present a wavelength allocation algorithm that requires at most $W \leq 2L_{\max} \log_2 N$ wavelengths for a ring with N nodes. We also prove a lower bound of $0.5L_{\max} \log_2 N$, thereby proving our algorithm to perform at most 4 times worse than the optimum. To the best of our knowledge, this is the first analytical result with provable bounds for this model.

An important conclusion from these results is that, at least as far as worst case analysis is concerned, fully dynamic scenarios result in significant degradation of the utilization of wavelengths, and that the difference between the efficiency of semi-dynamic scenarios and fully dynamic ones (i.e., the fact that deletions are allowed) grows logarithmically with the network size.

The paper is organized as follows. In Section 2 we address the static allocation problem, present the known upper bound algorithm and prove a tight lower bound for it. in Section 3 we address the dynamic problem, show that other known algorithms (circular first-fit, random allocation) fail to produce acceptable results, present our algorithm and analyze its worst-case performance. We also prove a lower bound which is only four times smaller than the upper bound. In Section 4 we suggest two improved algorithms and prove they are as good in the worst case as the original algorithm but are better if blocking is necessary, and in Section 5 we summarize the results and suggest further research.

2 Static Allocation

We start by considering the simplest case, in which the full set of lightpaths is given in advance. This case is applicable in many networks, in which the required set of lightpaths is determined as part of the network design phase of a higher level network.

The optimal algorithm in Figure 1 for this problem was suggested in [17]. A similar result is reported in [9]. Refer also to Figure 2 for a pictorial demonstration. This algorithm may require up to $W \leq L_{\max} + L_{\min}^{\text{node}}$ wavelengths, where $L_{\min}^{\text{node}}$ is defined as the minimum, over all *nodes*, of the number of lightpaths that go through the node (that is, are neither added nor dropped): up to $L_{\max}$ wavelengths

0. INPUT: a set of lightpath requests (a set of arcs on the ring).

1. Find a node v on the ring with a minimum number of routes traversing through it ($L_{\min}^{\text{node}}$).

2. Logically cut the ring at this point: duplicate v and split the route of each request passes that through v into two requests.

3. The resulting problem is that of allocating wavelengths to routes on a chain network. This problem is termed the *coloring problem of an interval graph*, and can be solved optimally (with $W = L_{\max}$ wavelengths) by a greedy algorithm [20, Sec.16.5] that assigns wavelengths to routes by scanning the chain from its leftmost point. The wavelengths are assigned in an arbitrary order: whenever the scan meets a new route, it assigns it one of the available wavelengths (which now becomes unavailable until the scan meets the rightmost end of the route).

4. When reconnecting the endpoints of the chain into the original ring, routes which were split at Step 2, may have different wavelengths assigned to their two parts. For each such case, assign a completely new wavelength to the route.

Figure 1: Static allocation of wavelengths

may be required for allocating lightpaths on the chain created in Step 2, and up to $L_{\min}^{\text{node}}$ for routes that have been split in Step 2 are assigned new wavelengths in Step 4. Furthermore, it is easy to see that $L_{\min}^{\text{node}} < L_{\max}$, since even if all links have a load of $L_{\max}$, not all the lightpaths that create this load can traverse through each and every node. Thus, $W \leq 2L_{\max} - 1$.

Next, we prove that in the worst case, indeed $W = 2L_{\max} - 1$ wavelengths are required.

Theorem 1. *Given a ring with $N > 2L_{\max}$ nodes, there exist lightpath request patterns that require $W = 2L_{\max} - 1$ wavelengths.*

Proof. Consider the set of requests depicted in Figure 3. These requests are divided into three groups: $\mathcal{A} = \{a_1, ..., a_{L-1}\}$, $\mathcal{B} = \{b_1, ..., b_{L-1}\}$ and $\{c\}$. All the routes in group $\mathcal{A}$ overlap on link A, and all the routes in group $\mathcal{B}$ overlap on link B. In addition, each $a_i \in \mathcal{A}$ overlaps all the $b_j \in \mathcal{B}$ for $j < i$ in the part of the ring below the line $[A, B]$, and all $b_j \in \mathcal{B}$ for $j \geq i$ above that line. In addition, c overlaps all the other routes. Thus, we have $2(L-1)+1$ routes that overlap each other and need at least $W = 2L - 1$ wavelengths. The maximal load is clearly $L_{\max} = L$.

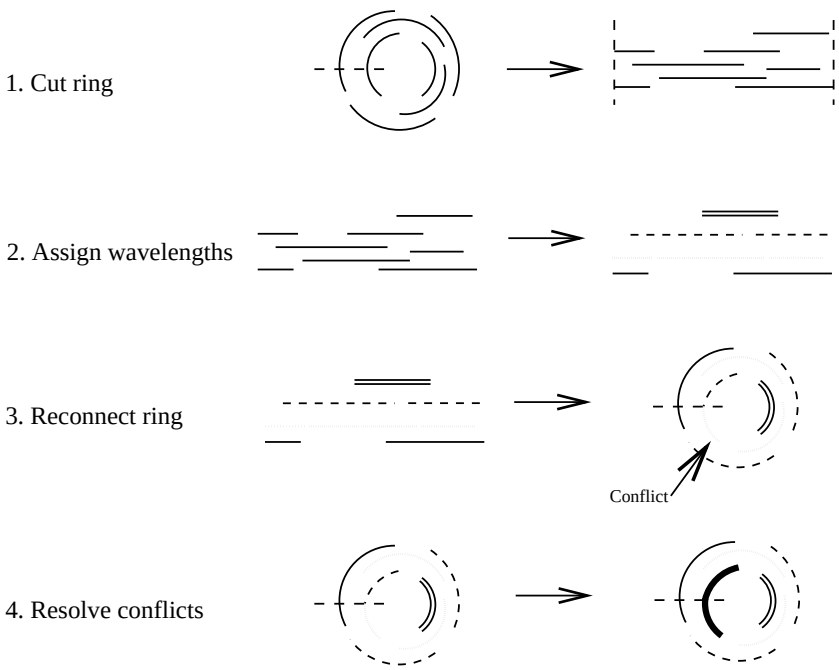


Figure 2: The phases of Figure 1

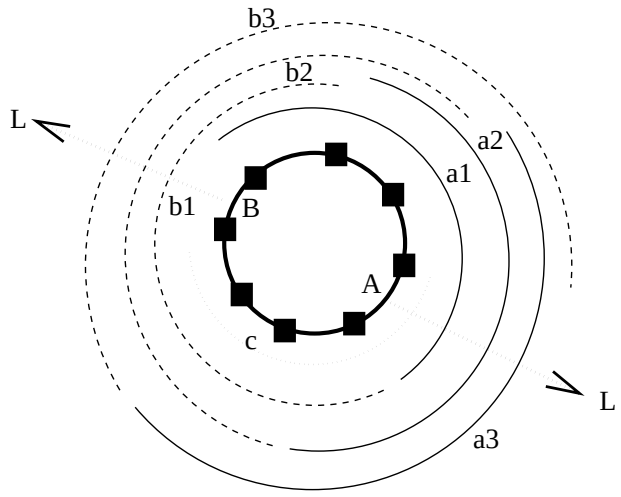


Figure 3: A worst case set of lightpath requests

More formally, number the nodes in the ring starting at an arbitrary node 0, and proceeding clockwise up to node $N-1$. Define

$$a_1 = [0, \frac{N}{2}], a_2 = [1, \frac{N}{2}+1], ..., a_i = [i-1, \frac{N}{2}+i-1], ..., a_L = [L-1, \frac{N}{2}+L-1],$$

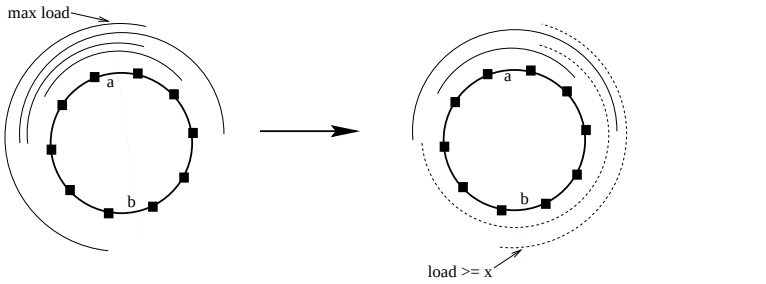
$$b_1 = [\frac{N}{2}, 1], b_2 = [\frac{N}{2}+1, 2], ..., b_i = [\frac{N}{2}+i-1, i], ..., b_L = [\frac{N}{2}+L-1, L],$$

$$c = [\frac{N}{2}-1, \frac{N}{2}+L-1].$$

Clearly the above arguments hold for this general case definition as well. □

3 Dynamic Allocation

In this section we discuss a different scenario, in which requests for lightpaths arrive at different times. We split the problem into two parts: first we show that the simple shortest path heuristics for determining the route of each lightpath request yields results which have up to twice the load of the optimum solution. Then, assuming that these routes for lightpath requests are given, we solve the wavelength allocation problem using up to four times more wavelengths than the best possible solution. We do



(a) Shortest path routing

(b) The routing after moving x of the routes to the other option around the ring

Figure 4: Shortest path routes are not much worse than optimal

not, however, assume any knowledge on the arrival/departure processes of these lightpaths. Thus, we achieve a robust guarantee for the performance of the system, without having to depend on assumptions which are all too often unjustified, especially when designing for future systems with no well-studied behavior.

3.1 Route determination

Consider a set of requests for lightpaths, for which only source and destination pairs are given for each request. In this section we prove that no algorithm can minimize the maximum load $L_{\max}$ more than a factor of two from the load created by the algorithm which routes each lightpath request on the shortest route between the source and destination.

Given any configuration of lightpaths (possibly after deletions of lightpaths) produced using shortest path routing, let L_{shrt} denote the maximum load ($L_{\max}$) for this case. Consider a link a with maximum load L_{shrt} depicted in Figure 4. Also consider the link b which is diametrically opposite to a on the ring. Since routes of lightpaths that cross a are the shortest possible, none of them crosses b as well (otherwise they would traverse more than half of the ring). Therefore, in any other solution that does not route x of them through a, these x are routed through b, and thus the load on b is at least x. It follows that the maximum load in any such solution cannot be reduced below $\frac{L_{\text{shrt}}}{2}$ of the load L_{shrt} used by the shortest path algorithm, by changing the routes of some of the requests to the other alternative around

the ring.

3.2 Fragmentation problem

In this section we address one of the main problems for dynamic scenarios, in cases when no reallocation of resources is possible. This problem is generally referred to as *fragmentation*, and is well studied in the context of computer memory management and disk management: after a long period of using a system, the free resource (free memory, unused disk space) becomes broken into small fragments, separated by small used segments, so that even if most of the resource is free, there is no sufficiently large contiguous fragment for a new request. This problem is easily solved if the system supports reallocation of resources. Thus, the small used memory segments that cause the fragmentation may be reallocated contiguously, leaving most of the free memory in one large piece. Unfortunately for our case, it is not desirable to reallocate wavelengths for lightpaths, since this involves disrupting the operation of very high bandwidth pipes.

The fragmentation problem is clarified by the following examples which show why the simple Circular-First-Fit and Random allocation algorithms fail to produce reasonable results. Namely, if the number of wavelengths, W, is as large as $NL_{\max}$, the chances for blocking are still very high. By contrast, our algorithm guarantees no blocking if $W = 2L_{\max}\log_2 N$.

3.2.1 Example 1: circular first fit allocation

In this example requests arrive in phases. In each phase, N requests arrive, each for a one hop lightpath, and are allocated different wavelengths (in a circular fashion)[1]. After $L_{\max} - 1$ phases of requests (each phase contains single hop requests that together traverses all the links), a new request arrives which includes all the ring nodes (or up to half the ring if shortest path routing is used). This new request has to be allocated a new wavelength. At this stage the configuration is the one described in Figure 5. Thus, the following theorem holds.

[1] If Circular-First-Fit is defined to try and allocate the previously chosen wavelength first (rather than the *next* avaliable wavelength), then the scenario described here can be simulated using additional deletions.

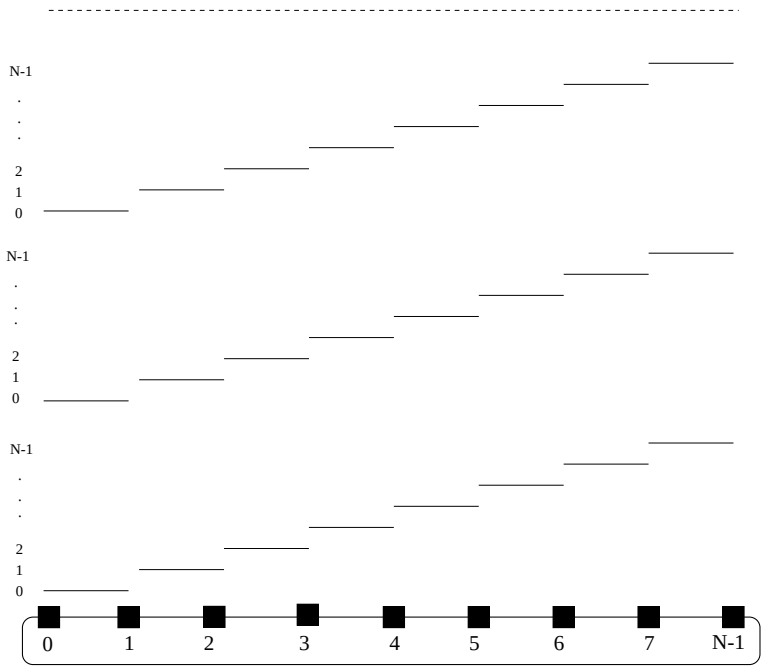


Figure 5: A worst case set of lightpath requests for the Circular-First-Fit algorithm

Theorem 2. *Given a ring with N nodes, and a set of lightpath requests with load $L_{\max}$, the Circular-First-Fit algorithm may need at least $W = 1 + N(L_{\max} - 1)$ wavelengths in the worst case to support all the requests.*

□

3.2.2 Example 2: Random allocation of wavelengths

A similar worst-case example holds for the case where the wavelength for a lightpath request is chosen at random among the available free wavelengths: Given a ring with W wavelengths per link and a set of $NL_{\max} - 1$ one hop lightpath requests, such that $L_{\max} - 1$ of them use each link, and then a single $N - 1$ hop lightpath request, X, we shall show that X will probably be blocked if $L_{\max}$ and N are large enough in comparison with W.

The probability of a specific wavelength w on a specific link ℓ being free after $L_{\max} - 1$ one hop lightpaths have been allocated on ℓ is clearly

$$p(w \text{ is free on link } \ell) = 1 - \frac{L_{\max} - 1}{W}.$$

Thus, the probability of w being free on all the $N - 1$ hops of X is

$$p(w \text{ is free for } X) = \left(1 - \frac{L_{\max} - 1}{W}\right)^{N-1}.$$

In order for X to be blocked, its route must be occupied on all wavelengths, thus

$$p(X \text{ is blocked}) = \left(1 - \left(1 - \frac{L_{\max} - 1}{W}\right)^{N-1}\right)^{W}.$$

Assuming $L_{\max} - 1 > \alpha \frac{W}{N}$ for some $\alpha > 0$ (or, in other words, if $W < \frac{1}{\alpha}(L_{\max} - 1)N$), and assuming N is large enough, we get

$$p(X \text{ is blocked}) \approx \left(1 - e^{-\alpha}\right)^{W}.$$

A few numerical examples that demonstrate the probability of request X to be blocked for a system of W wavelengths are concentrated in the following table.

$W \setminus \alpha$	1	1.5	2
2	39.9 %	60.3 %	74.7 %
4	15.9 %	36.4 %	55.8 %
8	2.5 %	13.2 %	31.2 %
16	0.065 %	1.76 %	9.7 %
32	4.2E-5 %	0.03 %	0.9 %

These examples demonstrate that for $W < (L_{\max} - 1)N$, the probability of X in above-mentioned scenario, to be blocked is unacceptably large for 16 wavelength systems (or less), and for $W < 0.5 L_{\max} N$ the probability is too large even for 32 wavelength systems.

3.3 Our algorithm

The DWLA-1 algorithm (Dynamic WaveLength Allocation, see Figure 7) allocates wavelengths in a dynamic setting with a performance guarantee of $W \leq 2L_{\max} \log_2 N$ for a ring with N nodes (as long

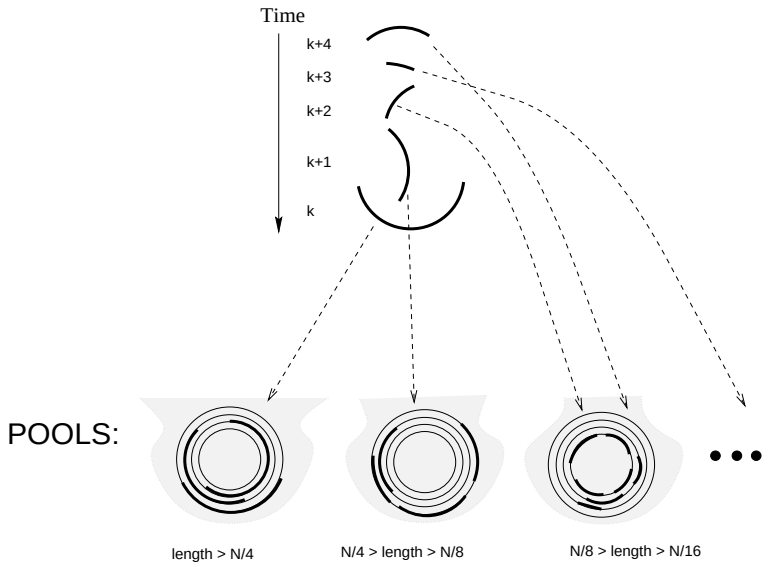


Figure 6: Operation of the dynamic case algorithm

as $L_{\mathbf{max}} \leq L_{\mathbf{alg}}$ for some predetermined value $L_{\mathbf{alg}}$). The key idea behind it is to avoid fragmentation by allocating each wavelength for routes of approximately the same length. Refer to Figure 6 for a pictorial demonstration of the algorithm in which requests fall down from the top of the figure and get sorted into the pools based on their length.

The main claim to be proven for this simple algorithm is that we do not run out of wavelengths in Step 3, as long as the load does not exceed the load for which the algorithm was designed, $L_{\mathbf{alg}}$.

Lemma 1. *Consider lightpaths of length* $N \cdot 2^{-i-1} \leq \ell(x) < N \cdot 2^{-i}$. *If the maximum load of such lightpaths does not exceed* $L_{\mathbf{alg}}$ *then* $2L_{\mathbf{alg}}$ *wavelengths suffice for them.*

Proof. Assume that the ring contains $N = 2^k$ nodes (the presentation of the proof is simplified by this assumption, and it is easy to see how other cases are dealt with). Consider two sets of links on the ring: one, $\mathcal{A} = \{A_j\}_{j=1}^{2^i}$, with gaps of size $N \cdot 2^{-i} = 2^{k-i}$ between each consecutive pair. The other, $\mathcal{B} = \{B_j\}_{j=1}^{2^i}$, having the same distance between each consecutive pair, and located in the middle of the gaps created by links in $\mathcal{A}$ (see Figure 8).

0. INPUT: receive add/delete lightpath requests, one at a time
   DATA STRUCTURE: Define a set of wavelength pools $\{\textsc{Pool}(i)\}_{i=0}^{\log_2 N-1}$, each pool containing $2L_{\text{alg}}$ wavelengths (L_{alg} being the anticipated maximum load).
1. If the request is to delete a lightpath, delete it and mark the relevant wavelength segment as free. Otherwise:
2. Let x be the current request to add a lightpath and let $\ell(x)$ be its length (the number of links traversed by the route). Choose i such that $N \cdot 2^{-i-1} \leq \ell(x) < N \cdot 2^{-i}$.
3. Find a free segment of a wavelength in pool $\textsc{Pool}(i)$ which can accommodate x, and allocate x on this wavelength.
4. Handle next request.

Figure 7: Dynamic allocation of lightpath requests (DWLA-1)

Given an lightpath x with length $N \cdot 2^{-i-1} \leq \ell(x) < N \cdot 2^{-i}$, it traverses no more than one A_x in $\mathcal{A}$. However, if it traverses no such link, it has to traverse one link of $\mathcal{B}$, say B_x. Thus, if x does not traverse any link in $\mathcal{A}$, it shares B_x with up to $L_{\text{alg}} - 1$ other lightpaths that do not traverse an $\mathcal{A}$ link, and L_{alg} wavelengths suffice for such lightpaths. On the other hand, if x does traverse A_x, then no more than additional $L_{\text{alg}} - 1$ lightpaths traverse A_x, proving that such lightpaths need no more than L_{alg} wavelengths. □

Theorem 3. *As long as $L_{\max} \leq \frac{W}{2\log_2 N}$ the DWLA-1 algorithm does not block any requests.*

3.4 Lower bound

We now prove that in the worst case $W \geq 0.5 L_{\max} \log_2 N$. We start with $L_{\max} = 2$.

Consider the following scenario, depicted in Figure 9. At each phase i, a request arrives for a lightpath that overlaps all the currently existing $i-1$ lightpaths. Thus any algorithm has to allocate it a new wavelength. Playing an adversary who issues the requests, we manage to manipulate any allocation algorithm (by means of additional add/delete requests) to utilize i wavelengths while the load $L_{\max}$ remains 2 at all times. This process can only be repeated $\log_2 N$ times, since in each phase i, the adversary is forced to issue lightpath requests traversing 2^i links. More formally, given some

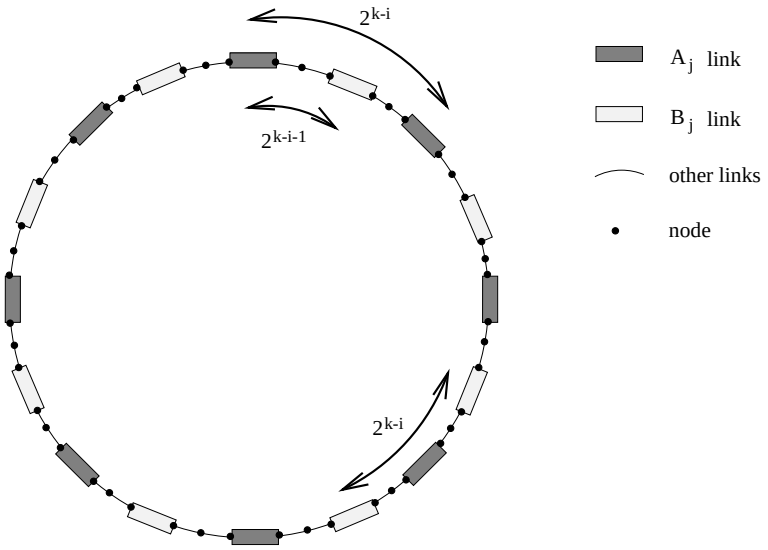


Figure 8: Chosen links on the ring

allocation algorithm Z, we now describe a worst case scenario specialized for it, in the following phases.

Phases 1 and 2. Two requests arrive to establish lightpaths p_1 and p_2 in the segment $[0, 1]$. Clearly they are allocated different wavelengths by Z.

Phase 3. A third request p_3 arrives for a lightpath in the segment $[1, 2]$. If Z allocates to it a wavelength which is different from those allocated to p_1 and p_2, then the phase ends — so far three wavelengths have been allocated. On the other hand, if Z allocates to p_3 the same wavelength that was allocated to either p_1 or p_2 (say p_1), then a request arrives for deleting p_1, and yet another lightpath addition request p_4 arrives for a lightpath in $[0, 2]$. Clearly Z allocates a third wavelength for p_4.

Phase 4. Phases 1–3 are repeated in the segment $[2, 4]$ as well. After which it is easy to see that it is possible to choose three non-overlapping lightpaths in segment $[0, 4]$ which have been allocated different wavelengths. For the rest of the lightpaths, delete requests arrive. Now, a new lightpath add request arrives for an lightpath in $[0, 4]$. Z has to allocate a new wavelength to it, resulting in a total of four different wavelengths. Note that $L_{\max}$ is still at most two.

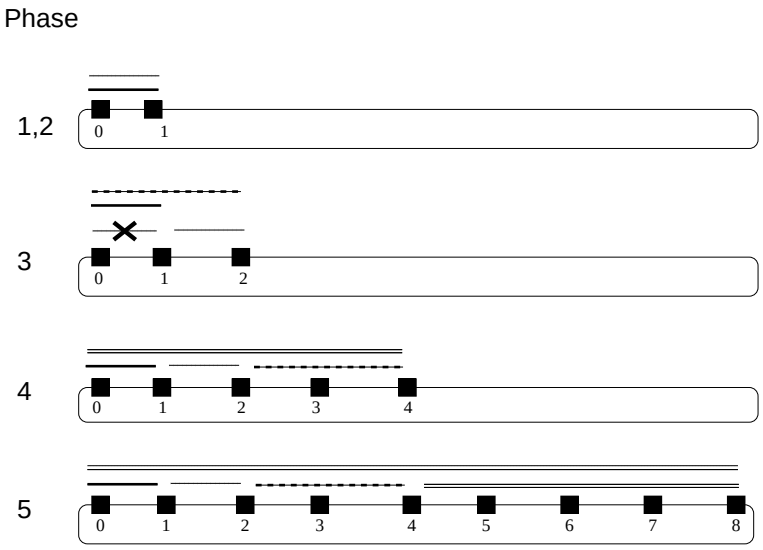


Figure 9: A worst case dynamic scenario of lightpath requests

$\vdots$

Phase i**.** After repeating Phases 1 to $i-1$ in segments $[0, 2^{i-1}]$ and $[2^{i-1}, 2^i]$, and deleting superfluous lightpaths to achieve a configuration of $i-1$ non-overlapping lightpaths of different wavelengths, a new request arrives to add an lightpath in the segment $[0, 2^{i-1}]$. Z allocates a new i^{th} wavelength to it, since it overlaps $i-1$ other wavelengths.

$\vdots$

Phase $\log_2 N$**.** The last lightpath request arrives in the segment $[0, N-1]$. Z allocates wavelength $\log_2 N$ to it.

This process required $W \geq \log_2 N$ wavelengths, with a maximum load of $L_{\max} = 2$. Thus, $W \geq 0.5 L \log_2 N$. To generalize the worst case to any (even) value of $L_{\max}$, we duplicate the number of arriving lightpath requests at each phase by $L_{\max}/2$. Since each of these $L_{\max}/2$ requests requires a different wavelength the whole allocation process is inflated by a factor of $L_{\max}/2$ wavelengths per phase, yielding the desired lower bound.

Theorem 4. *There exists some addition/deletion scenario that requires any wavelength allocation algorithm to use* $W > 0.5 L_{\max} \log_2 N$ *wavelengths.*

4 Improving the algorithm

As shown above, the DWLA-1 algorithm guarantees no blocking of requests if the load $L_{\max}$ does not exceed some value L_{alg}. However, it does not necessarily perform well if $L_{\max} > L_{\text{alg}}$. In such cases, some pool POOL(i) may be overflown and unable to accommodate additional requests of length $N \cdot 2^{-i-1} \leq \ell(x) < N \cdot 2^{-i}$, while other pools remain empty. While our lower bound shows that in the worst case there is no way to guarantee no blocking if the load is high, it is still desirable to minimize this blocking. The algorithms presented in this section dynamically adjust the pools so as to achieve better blocking if the load exceeds L_{alg}. Both of these algorithms are proven to work as good as DWLA-1 provided that the load is low enough ($L_{\max} < \frac{W}{2 \log_2 N}$). Hence they guarantee no blocking in this case. Both of them are also proven to block later than DWLA-1 if the load is higher, and indications are given as to why their blocking probability is lower. Simulation results to support the latter conjectures will be presented in the final version of the paper.

The modified algorithm of Figure 10 starts with empty pools (i.e., POOL(i) = $\emptyset$), and a new pool, FREE, which contains all free wavelengths (i.e., all wavelengths initially). Upon arrival of a request x of length $N \cdot 2^{-i-1} \leq \ell(x) < N \cdot 2^{-i}$, the algorithm tries to fit it into a wavelength of POOL(i). If no wavelength in the pool can accommodate x, a new wavelength is taken from FREE and added to POOL(i). Thus, each pool grows dynamically according to the needs of requests of the relevant length. Of course, when a wavelength is freed, it is returned to FREE.

Theorem 5. *The DWLA-2 algorithm never starts blocking requests earlier than DWLA-1 (and much later in most cases).*

Proof. Let x be the first blocked request in DWLA-2 and let POOL(i) be the wavelength pool for x. If POOL(i) in DWLA-2 is larger than L_{alg} (the size of POOL(i) in DWLA-1) then an earlier blocking event should have occurred in POOL(i) using DWLA-1. If POOL(i) is smaller than L_{alg} and FREE is empty,

0. INPUT: receive add/delete lightpath requests, one at a time
   DATA STRUCTURE: Define a set of wavelength pools $\{\text{POOL}(i)\}_{i=0}^{\log_2 N-1}$, each pool initially empty. Define an additional pool, FREE, containing all wavelengths.
1. If the request is to delete an lightpath, delete it and mark the relevant wavelength segment as free. If the wavelength is completely free, remove it from its current POOL(i) and return it to FREE. Goto Step 6.
2. Let x be the current request to add a lightpath and let $\ell(x)$ be its length (the number of links traversed by the route). Choose i such that $N \cdot 2^{-i-1} \leq \ell(x) < N \cdot 2^{-i}$.
3. Find a free segment of a wavelength in pool POOL(i) which can accommodate x.
4. If no such wavelength exists in POOL(i), add a new wavelength from FREE to POOL(i) (and remove it from FREE).
5. Allocate x on the free segment of the abovementioned wavelength.
6. Handle next request.

Figure 10: First improved dynamic allocation of lightpath requests (DWLA-2)

than some other pool, POOL(j) is larger than L_{alg}. This is due to some earlier request that expanded POOL(j) beyond L_{alg}. Thus, this earlier request would have been blocked by DWLA-1. □

Corollary 1. *If $L_{\max} \leq \frac{W}{\log_2 N}$ then DWLA-2 guarantees no blocking.*

DWLA-2 typically delays the first blocking event much more than DWLA-1 since the bounds between pools are not fixed. Thus, when the length of the requests, $\ell(x)$, is unevenly distributed, blocking occurs only when FREE is empty, and not when the specific pool is saturated. For example, if all requests are single hop requests ($\ell(x) = 1$), and all of them occur between the same pair of adjacent nodes on the ring, then DWLA-1 will block after $\frac{W}{\log_2 N}$ requests, since POOL($\log_2 N$) will be full. However, DWLA-2 will block only after W requests.

DWLA-2 still causes one main problem, which does not allow it to exploit the wavelength resource efficiently enough if the load is high. This problem is analogous to the "trunking effect" encountered in conventional telecommunication systems: The reservation of resources to perform more focused tasks increases the overall blocking. The effect occurs here when lightpaths with a long life-span keep

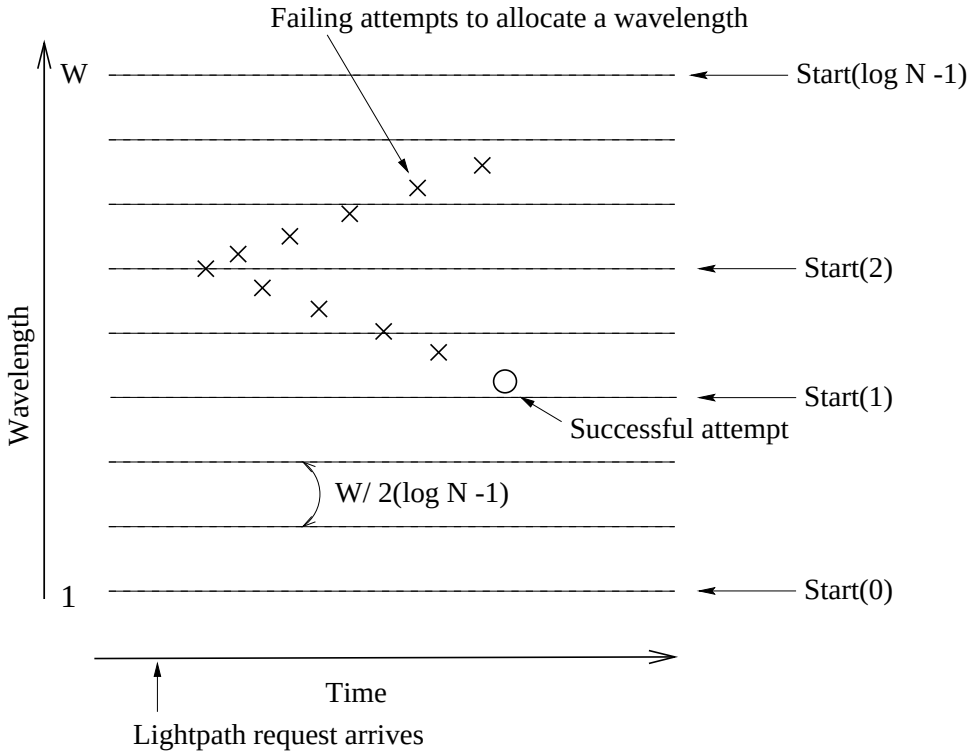


Figure 11: Second improved dynamic algorithm (DWLA-3)

wavelengths from returning to the FREE pool, since as long as there is a lightpath that uses a wavelength, this wavelength remains in the specific POOL(i) pool and other segments of it can not be reused for lightpaths of very different length.

To alleviate this problem, a third algorithm, DWLA-3, is introduced. This algorithm is a generalization of the Incr/Decr algorithm of [12]. The algorithm starts allocating wavelengths for a request of length $\ell(x)$, at wavelength START(i). If START(i) cannot accommodate the request, DWLA-3 searches an available wavelength in the next/previous wavelengths (START(i) $\pm$ 1), and increases the distance from START(i) until a free wavelength is found (see Figure 11). For a more formal description refer to Figure 12.

Similarly to the DWLA-2 case, we can prove the following.

0. INPUT: receive add/delete lightpath requests, one at a time

   DEFINE: $\textsc{Start}(i) = \begin{cases} 1 & i = 0, \\ W & i = \log_2 N - 1, \\ 1 + \frac{(2i+1)W}{2(\log_2 N - 1)} & \text{otherwise.} \end{cases}$

1. If the request is to delete an lightpath, delete it and mark the relevant wavelength segment as free.

2. Let x be the current request to add a lightpath and let $\ell(x)$ be its length (the number of links traversed by the route). Choose i such that $N \cdot 2^{-i-1} \leq \ell(x) < N \cdot 2^{-i}$.

3. Let $\Delta \leftarrow 0$.

4. If one of the wavelengths $\textsc{Start}(i) \pm \Delta$ can accommodate x, allocates it and finish (goto Step 6).

5. If not, $\Delta \leftarrow \Delta + 1$ and goto Step 4.

6. Handle next request.

Figure 12: Second improved dynamic allocation of lightpath requests (DWLA-3)

Theorem 6. *The DWLA-3 algorithm never starts blocking requests earlier than DWLA-1 (and much later in most cases).*

Corollary 2. *If* $L_{\max} \leq \frac{W}{\log_2 N}$ *then DWLA-3 guarantees no blocking.*

This algorithm is expected to have even better blocking probability since no wavelength is permanently allocated to lightpaths of a given length range. On the other hand, in most cases the majority of lightpaths of length $N \cdot 2^{-i-1} \leq \ell(x) < N \cdot 2^{-i}$ will be allocated close to wavelength $\textsc{Start}(i)$, thereby decreasing the fragmentation.

5 Summary and further research

In this paper we have studied the problem of allocating wavelengths to lightpaths in a WDM ring system in which wavelength conversion is not possible. We have first shown that the case in which all lightpath requests are known in advance enables to design an algorithm for allocating wavelengths which uses less wavelengths than twice the maximum load. While the algorithm is known in the literature, we have

shown in the current paper that in the worst case no better algorithm exists. For the dynamic case, we have suggested an algorithm which uses up to $2L_{\max} \log_2 N$ wavelengths. Another way to look at this result is a guarantee of no blocking as long as the load does not exceed $\frac{W}{2 \log_2 N}$. We have also shown that in the worst case this algorithm performance is up to four times the minimum possible number of wavelengths that any algorithm needs.

We have suggested two improved algorithms, DWLA-2 and DWLA-3, which have the same guarantees if the load is low enough, but have delayed and better blocking probability if the load is higher. While we have indicated why these algorithms should perform better than DWLA-1, it is still necessary to demonstrate this empirically, and such result will appear in the final version.

An interesting remaining issue is to find an algorithm with better worst case performance. However, for deployment in real implementations, the average performance of the algorithm must be studied as well and should prove competitive with other allocation algorithms.

Acknowledgment. We would like to thank Rajiv Ramaswami for the fruitful discussions, in particular for Section 3.1.

References

[1] B. Mukherjee, S. Ramamurthy, D. Banerjee, and A. Mukherjee, "Some principles for designing a wide-area optical network," in *IEEE Infocom'94*, 1994.

[2] I. Chlamtac, A. Ganz, and G. Karmi, "Lightnets: Topologies for high-speed optical networks," *IEEE/OSA Journal on Lightwave Technology*, vol. 11, pp. 951–961, May/June 1993.

[3] A. Aggarwal, A. Bar-Noy, D. Coppersmith, R. Ramaswami, B. Schieber, and M. Sudan, "Efficient routing and scheduling algorithms for optical networks," in *Proc. 5th Annual ACM-SIAM Symp. on Discrete Algorithms*, pp. 412–423, Jan. 1994.

[4] S. V. Jagannath, K. Bala, and M. Mihail, "Hierarchical design of WDM optical networks for ATM transport," *IEEE Globecom 1995*, 1995.

[5] R. L. Cruz, G. R. Hill, A. L. Kellner, R. Ramaswami, and G. H. Sasaki, eds., *IEEE JSAC/ IEEE/OSA JLT: Special Issue on Optical Networks*, June 1996.

[6] K. Nosu, H. Toba, K. Inoue, and K. Oda, "100 channel optical fdm technology and its applications to optical FDM channel-based networks," *IEEE/OSA Journal on Lightwave Technology*, vol. 11, pp. 764–776, May/June 1993.

[7] A. Frank, T. Nishizeki, N. Saito, H. Suzuki, and E. Tardos, "Algorithms for routing around a rectangle," *Discrete Applied Math*, vol. 40, pp. 363–378, 1992.

[8] M. Garey, D. Johnson, G. Miller, and C. Papadimitriou, "The complexity of coloring circular arcs and chords," *SIAM Journal on Discrete Math*, vol. 1, no. 2, pp. 216–227, 1980.

[9] P. Raghavan and E. Upfal, "Efficient routing in all-optical networks," in *Proc. 26th ACM Symp. Theory of Computing*, pp. 134–143, May 1994.

[10] M. Mihail, C. Kaklamanis, and S. Rao, "Efficient access to optical bandwidth," in *36th Symp. on Foundations of Computer Science*, pp. 548–557, 1995.

[11] A. Birman, "Computing approximate blocking probabilities for a class of optical networks," *IEEE JSAC/JLT Special Issue on Optical Networks*, June 1996.

[12] A. Birman and A. Kershenbaum, "Routing and wavelength assignment methods in single-hop all-optical networks with blocking," in *IEEE Infocom'95*, pp. 431–438, 1995.

[13] I. Chlamtac, A. Ganz, and G. Karmi, "Lightpath communications: an approach to high-bandwidth optical WAN's," *IEEE Transactions on Communications*, vol. 40, pp. 1171–1182, July 1992.

[14] M. Kovačević and A. Acampora, "On the benefits of wavelength translation in all-optical clear-channel networks," *IEEE JSAC/JLT Special Issue on Optical Networks*, June 1996.

[15] R. Ramaswami and K. Sivarajan, "Routing and wavelength assignment in all-optical networks," *ACM/IEEE Transactions on networking*, pp. 489–500, Oct. 1995. An earlier version appeared in Infocom'94.

[16] N. Waters and P. Demeester, "Design of the optical path layer in multiwavelength cross-connected networks," *IEEE JSAC/JLT Special Issue on Optical Networks*, June 1996.

[17] A. Tucker, "Coloring a family of circular arcs," *SIAM Journal on Applied Math*, vol. 29, no. 3, pp. 493–502, 1975.

[18] M. Ślusarek, "Optimal online coloring of circular arc graphs," *Informatique Theoretique et Applications*, vol. 29, no. 5, pp. 423–429, 1995.

[19] H. Kierstead and W. Trotter, "An extremal problem in recursive combinatorics," *Congressus Numerantium*, vol. 33, pp. 143–153, 1981.

[20] C. Berge, *Graphs and Hypergraphs*. North Holland, 1976.