# Approximation Algorithms for Routing and Call Scheduling in All-Optical Chains and Rings*

Luca Becchetti[1]**, Miriam Di Ianni[3], and Alberto Marchetti-Spaccamela[2]

[1] Technische Universität Graz, Institut für Mathematik B;
luca@opt.math.tu-graz.ac.at
[2] Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza";
alberto@dis.uniroma1.it
[3] Dipartimento di Ingegneria Elettronica e dell'Informazione, Università di Perugia;
diianni@diei.unipg.it

**Abstract.** We study the problem of routing and scheduling requests of limited durations in an all-optical network. The task is servicing the requests, assigning each of them a starting time and a wavelength, with restrictions on the number of available wavelengths. The goal is minimizing the overall time needed to serve all requests. We propose constant approximation algorithms for both ring and chain networks. In doing this, we also propose a polynomial-time approximation scheme for the problem of routing weighted calls on a directed ring with minimum load.

## 1 Introduction

*All-optical networks* allow very high transmission rates, widely exceeding those that can be guaranteed by traditional electronic technology. Wavelength Division Multiplexing (WDM) allows the concurrent transmission of multiple data streams on the same optic fiber; different data streams on the same optical link at the same time and in the same direction use different wavelengths (currently 30-40 in experimental settings) [6,13]. Moreover, the high speed achievable with all-optical networks is mainly due to the fact that the signal is kept in optical form throughout its transmission from source to destination.

In this paper, we address the problem of Call Scheduling in all-optical networks, that is the problem of scheduling a set of communication requests (*calls*) each one characterized by a source-destination pair and a duration. Following [13] we assume that optical links allow only one way communication: if there is a link from $x$ to $y$ then the information flow is unidirectional from source to tail. In order to allow bidirectional connection, we assume that if there is a link from $x$ to $y$ there is also a link from $y$ to $x$ [17]. A different model assumes that the optical links are undirected and allow bidirectional communication; we remark that this model is less realistic with respect to current technology [13].

* This work was partially supported by ESPRIT project ALCOM-IT and by the START program Y43-MAT of the Austrian Ministry of Science.

** Part of this research was done while the author was at Dipartimento di Informatica e Sistemistica, University of Rome "La Sapienza".

C. Pandu Rangan, V. Raman, R. Ramanujam (Eds.): FSTTCS'99, LNCS 1738, pp. 201–213, 1999.

Given a network and a set of calls MINIMUM CALL SCHEDULING requires to assign a directed path, a wavelength and a starting time to each call, subject to the constraint that no pair of calls assigned to the same wavelength use the same (directed) arc at the same time and that the number of assigned wavelength does not exceed some bound $k$. The objective is to minimize the overall time to accomodate all calls.

**Related work**. To the best of our knowledge, call scheduling in all-optical networks received so far little attention.

Concerning optical networks, several related problems have been extensively studied. MIN PATH COLORING [3,4,15,17] aims to find routes and a wavelength assignment for a set of calls of infinite duration that minimize the number of used wavelengths. In the MAX CALL ADMISSION problem calls are presented in an on-line fashion; when a call arrives it can be either rejected or accepted; in the latter case it must be immediately satisfied using available resources. The objective is maximizing the number of accepted calls [2,3].

In the MIN RING LOADING problem, arising in the project of SONET networks [18], the aim is that of devising a routing of calls in a ring network such that the maximum load of a link (defined as the sum of the durations of all calls that use the link) is minimized. This problem is polynomial-time solvable under the assumption that all calls have unit durations [19,18]. The undirected ring loading problem (i.e., each link allows two ways communication) has been considered in [18], where a constant approximation algorithm is presented. This result has been improved and a polynomial time approximation scheme has been proposed [14]. The case of links that allow one way communication is the MINIMUM WEIGHTED DIRECTED RING ROUTING (MIN-WDRR) problem and will be considered in the present paper.

Scheduling calls with minimum makespan has been considered in non optical networks. Concerning packet switched networks, a seminal result by Leighton [16] proved the existence of a schedule delivering all packets in a number of steps within a constant factor of the lower bound. In [7,8] the call scheduling problem has been studied in the context of ATM networks. In particular, the authors propose approximation algorithms for star and ring networks.

**Results of the paper**. In section 3 we propose approximation algorithms for call scheduling in chain networks. Namely we propose a 3-approximation algorithm for the call scheduling problem in chains when just one wavelength is available, by a reduction to the problem of MINIMUM DYNAMIC STORAGE ALLOCATION (MIN-DSA) for which a 3-approximation algorithm is known [11]. Successively, we extend the algorithm to the general case of $k$ available wavelengths, obtaining a 5-approximation algorithm. The above results hold also in the case of undirected chains.

In section 4 we first give a polynomial-time approximation scheme for MINIMUM WEIGHTED DIRECTED RING ROUTING. This result, together with those of Section 3, allows to obtain a $(12 + \epsilon)$-approximation algorithm for the call scheduling problem in ring networks. We remark that the problem of routing

weighted calls in a ring network with the aim of minimizing the maximum load, has itself a practical relevance in SONET networks [18,14,19].

The results of section 3 hold also in the case of undirected chains. For the sake of brevity, most proofs are omitted.

## 2 Preliminaries

In the following, $G = (V, E)$ denotes a simple, directed graph on the vertex set $V = \{v_0, v_1, \ldots, v_{m-1}\}$ such that arc $(v_i, v_j)$ exists *if and only if* arc $(v_j, v_i)$ exists. A *network* is a pair $\langle G, k\rangle$ where $k$ is the number of available wavelengths (from now on, colors) on each arc. A *call* $C = [s, d, l]$ is an ordered pair of vertices $s, d$ completed by an integer $l > 0$ representing the call duration. Namely, $s$ is the source vertex originating the data stream to be sent to the destination $d$.

Given a network $\langle G, k\rangle$ and a set $\mathcal{C} = \{C_h = [s_h, d_h, l_h] : h = 1, \ldots, n\}$ of calls, a *routing* is a function $R : \mathcal{C} \to \mathcal{P}(G)$, where $\mathcal{P}(G)$ is the set of simple paths in $G$. Given a network $\langle G, k\rangle$, a set $\mathcal{C}$ of calls and a routing $R$ for $\mathcal{C}$, a *schedule* is an assignment of starting times and colors to calls such that at every time no two calls with the same color use the same arc; formally, a schedule is a pair $\langle S, F\rangle$, with $S : \mathcal{C} \to [0, 1, \ldots \sum_{h=1}^{n} l_h]$ a function assigning starting times to calls and $F : \mathcal{C} \to \{1, \ldots k\}$ a function assigning colors to calls. $S$ and $F$ must be such that, for any $(u, v) \in E$ and $C_i, C_j \in \mathcal{C}$, if $(u, v) \in R(C_i)$, $(u, v) \in R(C_j)$ and $F(C_i) = F(C_j)$ then either $S(C_i) \geq S(C_j) + l_j$ or $S(C_j) \geq S(C_i) + l_i$. $T = \max_{1 \leq h \leq n}\{S([s_h, d_h, l_h]) + l_h\}$ is the *makespan* of schedule $\langle S, F\rangle$. $T^*(x)$ or simply $T^*$ denotes the makespan of an optimal solution for an instance $x$..

In the Minimum Call Scheduling (min-cs) problem it is required to find a routing and a schedule for an instance $\langle\langle G, k\rangle, \mathcal{C}\rangle$ having minimum makespan. Since the problem is trivial if $k \geq |\mathcal{C}|$, we assume $k < |\mathcal{C}|$. Notice that, if routes are fixed, the problem of scheduling calls in order to minimize the makespan is unapproximable within $O(|\mathcal{C}|^{1-\delta})$, for any $\delta > 0$ [5] in general networks.

Given an instance of min-cs and a routing $R$, the *load* $L_R(e)$ of arc $e$ is defined as the sum of the durations of calls using it. If $k = 1$, then $L^* = \min_R(\max_e L(e))$ is a natural lower bound to the makespan of any schedule.

In the following we show that min-cs in chain networks is closely related to Minimum Dynamic Storage Allocation (min-dsa), that requires the allocation of contiguous areas in a linear storage device in order to minimize the overall requested storage space. A *block* $B = (f, e, z)$ represents a storage requirement such that $f$ and $e$ are, respectively, the first and last time instants in which block $B$ must be allocated and $z$ is an integer denoting the memory requirement (size) of $B$. An instance of min-dsa consists of a set of blocks $\mathcal{B} = \{B_1 = (f_1, e_1, z_1), \ldots, B_n = (f_n, e_n, z_n)\}$; an *allocation* of $\mathcal{B}$ is a function assigning blocks to storage locations, such that if both $B_i$ and $B_j$ must stay at the same time in the storage device then they must be assigned to non overlapping portions. Formally, an allocation is a function $F : \mathcal{B} \to N$ such that, for any pair $B_i, B_j$ of blocks, if $f_i \leq f_j \leq e_i$ or $f_j \leq f_i \leq e_j$ then either $F(B_i) + z_i - 1 < F(B_j)$ or $F(B_j) + z_j - 1 < F(B_i)$. min-dsa is defined as follows: given an infinite

array of storage cells and a set $\mathcal{B}$ of blocks, find an allocation $F$ such that the storage size $M = \max_{1 \le h \le n}\{F(B_h) + z_h\}$ is minimum. MIN-DSA is NP-hard [9] but approximable within a constant [10,11].

## 3 Call Scheduling in Chain Networks

In this section we assume that the network is a chain with $m$ vertices, simply denoted as $0, 1, \ldots, m-1$. The restriction of MIN-CS to chain networks will be denoted as CHAIN-MIN-CS (or CHAIN-MIN-CS$_k$, when $k$ is the number of available colors). Since the network is a chain, there is only one path between each source-destination pair. This implies that routing of calls is fixed and that the set $\mathcal{C}$ of calls consists of two independent subsets $\mathcal{C}_1 = \{[s_{11}, d_{11}, l_{11}], \ldots, [s_{1n_1}, d_{1n_1}, l_{1n_1}]\}$ and $\mathcal{C}_2 = \{[s_{21}, d_{21}, l_{21}], \ldots, [s_{2n_2}, d_{2n_2}, l_{2n_2}]\}$, where $n_1 + n_2 = n$ and, for any $h = 1, \ldots, n_1$, $s_{1h} < d_{1h}$ while, for any $h = 1, \ldots, n_2$, $s_{2h} > d_{2h}$. This implicitely defines two independent subinstances for each instance of the problem. Without loss of generality, in the following we always refer to just the first of them. The main result of this section is a 5-approximation algorithm for MIN-CS. This result exploits a close relationship between MIN-CS$_1$ and MIN-DSA, which is stated in the next theorem:

**Theorem 1.** *There exists a polynomial-time reduction from* CHAIN-MIN-CS$_1$ *to* MIN-DSA *such that an instance of the first problem admits a schedule with makespan $T$ if and only if the corresponding instance of the second problem admits a storage allocation with size $T$.*

The reverse reduction also holds, but we do not need it here. Our first approximation result is a direct consequence of theorem 1 and the 3-approximation algorithm for MIN-DSA given in [11].

**Corollary 1.** *There exists a 3-approximation algorithm for chain-*MIN-CS$_1$.

We now turn to the general case. The algorithm we propose is sketched below: the first step rounds up call durations to the closest power of 2; this worsens the approximation of the makespan of the optimum schedule by a factor at most 2.

**Algorithm** *chain-cs*
**input:** chain graph $G$, set $\mathcal{C}$ of calls, integer $k$;
**begin**
1. **for** $h := 1$ to $n$ **do** Round $l_h$ to the closest upper power of 2;
2. Find a pseudo-schedule assuming only 1 available color;
3. Assign calls to colors;
4. Find a schedule separately for each color;

**end**.

We assume that call durations are powers of 2. Assuming only one color, in the sequel we will use the following interpretation: a call $C = [s, d, l]$ scheduled in interval $[t, t + l]$ can be graphically represented as a rectangle of height $l$,

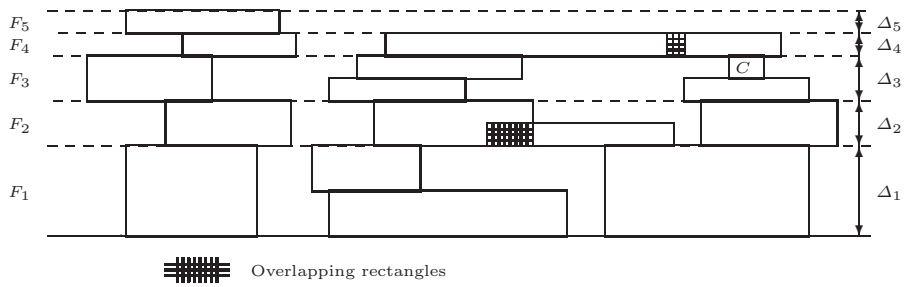


**Fig. 1.** Example of pseudo-schedule for 1 color

whose $x$-coordinates vary in $[s, d]$ and whose $y$-coordinates vary in $[t, t + l]$. We now describe steps 2-4 more in detail.

**Find a pseudo-schedule**.

We first determine a pseudo-schedule for $\mathcal{C}$, assuming only 1 color is available. A *pseudo-schedule* is an assignment $PS : \mathcal{C} \rightarrow \{0, \ldots \sum_{h=1}^{n} l_h\}$ of starting times to calls, such that each arc may be used by more than one call at the same time. The *length* of $PS$ with respect to the set $\mathcal{C}$ of calls is $\max_{h \in \{1,\ldots,n\}}\{PS(C_h)+l_h\}$.

To our purposes, we need a pseudo-schedule in which at most two calls may use the same arc at the same time, based on the following definition: assume that, for some $h \leq n - 1$, we have a pseudo-schedule for calls $C_1, \ldots, C_h$; we say that $C_{h+1}$ is *stable* [10] at time $t$ with respect to the pseudo-schedule of $C_1, \ldots, C_h$ if and only if and only if there exists an arc $e$ such that $C_{h+1}$ uses $e$ and, for every instant $0, \ldots, t-1$, $e$ is used by at least one call in $\{C_1, \ldots, C_h\}$. In order to obtain such a pseudo-schedule, we proceed inductively as follows: i) calls are first ordered according to non-increasing durations (i.e. if $j > i$ then $l_i \geq l_j$); ii) $C_1$ is assigned starting time 0; iii ) assuming $C_1, \ldots, C_h$ have each been assigned a starting time, $C_{h+1}$ is assigned the maximum starting time such that it is stable with respect to the pseudo-schedule of $C_1, \ldots, C_h$.

Under the assumption that call durations are powers of 2 it is possible to show [10] that this choice leads to a pseudo-schedule in which at most two calls use the same arc at the same time. An example of a pseudo-schedule is presented in Figure 1 where, as described above, each call $C_h = [s_h, d_h, l_h]$ corresponds to a rectangle of height $l_h$ whose horizontal coordinates vary between $s_h$ and $d_h$.

In the following we use $PS$ to denote the particular pseudo-schedule constructed above. Note that the notion of stability implies that the length of $PS$ is a lower bound to the makespan of the optimal schedule for one color.

**Assign calls to colors**.

We partition the set of calls in the pseudo-schedule into $\gamma$ subsets $F_1, F_2, \ldots F_\gamma$ called *stripes*, by "slicing" the pseudo-schedule into horizontal stripes (figure 1).

Stripes are defined inductively as follows: i) let $C_{h_1}$ be the longest call starting at time 0 in the pseudo-schedule. Then stripe $F_1$ has *height* $\Delta_1 = l_{h_1}$ and includes all calls that in $PS$ start not after $\Delta_1$; ii) assume that stripes $F_1, \ldots, F_{i-1}$ have

been defined ($i \geq 2$) and that there are still calls not assigned to a stripe, let $t$ be the time at which the last call in $F_{i-1}$ ends and let $C_{h_i}$ be the longest call starting at $t$ in the pseudo-schedule $PS$. Then stripe $F_i$ has *height* $\Delta_i = l_{h_i}$ and includes all calls that start not before $t$ and end not after $t + l_{h_i}$. Figure 1 illustrates an example of slicing of a pseudo-schedule into stripes.

Let $\mathcal{F} = \{F_1, F_2, \ldots F_\gamma\}$ be defined by the construction given above. The next lemma states that $\mathcal{F}$ defines a partition of $\mathcal{C}$

**Lemma 1.** *Every call in $\mathcal{C}$ belongs to one and only one set in $\mathcal{F}$.*

Stripes (and the corresponding calls) are then assigned to colors by defining a proper instance $x$ of MULTIPROCESSOR SCHEDULING with $k$ identical machines with the goal of minimizing the total makespan. Roughly, jobs of instance $x$ correspond to the sets $\{F_1, F_2, \ldots F_\gamma\}$ of the partition $\mathcal{F}$ obtained above. Namely, with each stripe of height $\Delta$ we associate a job of the same size.

We use the LPT (*Longest Processing Time* first) rule proposed by Graham [12] to solve the obtained instance of MULTIPROCESSOR SCHEDULING: jobs are first ordered according to decreasing sizes and then greedily assigned to the processors (i.e. a job is assigned to the less heavily loaded processor). The obtained solution yields an assignment of calls to colors, defined as follows: if stripe $F_i$ is assigned to machine $j$, then all calls in $F_i$ are assigned color $j$. Now, for each color $j$, we derive a pseudo-schedule $PS_j$ of the calls that have been assigned color $j$ in the following way: assume machine $j$ has been assigned stripes $F_{j_1}, \ldots, F_{j_r}$ (in this order) and let $C$ be a call assigned to color $j$ and belonging to stripe $F_{j_p}$, $p \in \{1, \ldots, r\}$. Recall that $PS(C)$ is the starting time of $C$ in the pseudo-schedule obtained in step 2 of *chain-cs*. The starting time $PS_j(C)$ of $C$ in the pseudo-schedule for color $j$ is given by the sum of the heights of stripes $F_{j_1}, \ldots, F_{j_p-1}$ plus the offset of $C$ in its stripe $F_{j_p}$. Namely, we have that $PS_j(C) = \sum_{i=1}^{p-1} \Delta_{j_i} + PS(C) - \sum_{i=1}^{j_p-1} \Delta_i$.

Figure 2 illustrates the effect of *Assign-Calls-To-Colors* over the pseudo-schedule of figure 1 in the case of 2 colors. In particular, call $C \in F_3$ in figure 1 has been assigned color 2 and $PS_2(C) = \Delta_2 + PS(C) - (\Delta_1 + \Delta_2)$.

We now analyze the makespan of the pseudo-schedule. The assumption that the duration of each call is a power of 2 and the characteristics of the pseudo-schedule obtained in the first step of the algorithm allow a tight analysis of LPT, yielding the next Lemma. For any color $j = 1, \ldots, k$, let $T^j$ be the length of the associated pseudo-schedule after procedure *Assign-Calls-To-Colors* and let $T_{PS} = max_j T^j$ (see fig. 2). Finally, let $T^*$ denote the makespan of an optimal schedule for the instance under consideration.

**Lemma 2.** $T_{PS} \leq T^*$.

**Find a Schedule**.
For every color $j = 1, \ldots, k$, let $PS_j$ denote the pseudo-schedule obtained after procedure *Assign-Calls-To-Colors* has been run and let $\mathcal{C}_j$ denote the set of calls that are assigned color $j$. It remains to derive a proper schedule $S_j$ from $PS_j$, for any color $j = 1, \ldots, k$. The relationship between MIN-DSA and CHAIN-MIN-CS$_1$ stated in theorem 1 implies the following lemma.

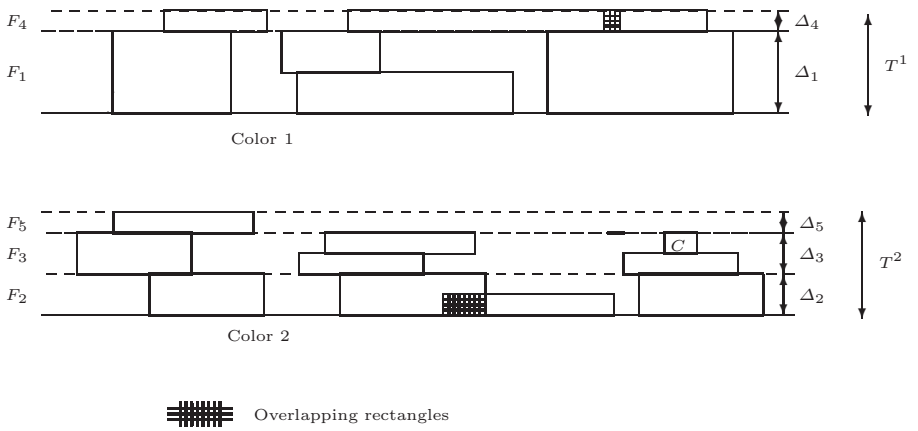


**Fig. 2.** Example of pseudo-schedule for 2 colors ($T_{PS} = T^1 = T^2$ in this case)

**Lemma 3.** *[10] If call durations are powers of 2 then there exists an algorithm for scheduling calls in $PS_j$ with makespan at most 5/2 times the length of $PS_j$.*

Lemmas 1, 2 and 3 imply the following theorem.

**Theorem 2.** *Algorithm* chain-min-cs *finds a schedule whose makespan is at most 5 times the optimum.*

Notice that the same result above holds also in the case of undirected chains.

## 4 Call Scheduling in Ring Networks

In the sequel, $\Sigma_m$ denotes a directed ring with $m$ vertices, clockwise indexed $0, \ldots, m-1$. The arc set is $\{(v, (v+1) \bmod m), (v, (v-1) \bmod m) : 0 \leq i \leq m-1\}$, where $(v, (v+1) \bmod m)$ (respectively $(v, (v-1) \bmod m)$) denotes the clockwise (counter clockwise) directed arc between vertices $v$ and $(v+1) \bmod m$ ($v$ and $(v-1) \bmod m$). In the following we perform operations modulo $m$ and we write $v+1$ ($v-1$) for $(v+1) \bmod m$ (respectively $(v-1) \bmod m$). We define $[s, t]$ to be $\{u | s \leq u \leq t\}$ if $s \leq t$, $[s, m-1] \cup [0, t]$ otherwise.

A call in a ring can be routed clockwise or counter clockwise. We first find a routing of the calls minimizing the load and then we schedule them by using a constant approximation algorithm based on the results of Section 3.

### 4.1 Routing with Minimum Load

Given a directed ring $\Sigma_m$ and a set $\mathcal{C}$ of calls, Minimum Weighted Directed Ring Routing (min-wdrr) is the problem of routing calls in $\mathcal{C}$ so that the maximum load on the arcs is minimized. $A_v$ ($B_v$), $v = 0, 1, \ldots, m-1$, denotes

the load on arc $(v, v+1)$ (arc $(v+1, v)$). With each call $C_h \in \mathcal{C}$ we can associate a binary variable $x_h$ such that $x_h = 1$ if $C_h$ is routed clockwise, $x_h = 0$ otherwise. Under these assumptions, MIN-WDRR can be formally described as follows:

$\min L = \max\{\max_v A_v, \max_v B_v\}$

$$A_v = \sum_{h:[v,v+1]\subseteq[s_h,t_h]} l_h x_h, \quad B_v = \sum_{h:[v+1,v]\subseteq[t_h,s_h]} l_h(1-x_h), \quad x_h \in \{0,1\}, 1 \le h \le n.$$

$OPT$ denotes the value of an optimal solution for any given instance of MIN-WDRR. MIN-WDRR is polynomial-time solvable when $l_h = 1$, $h = 1, 2, \ldots, n$ [19].

**Theorem 3.** MIN-WDRR *is NP-hard.*

**Lemma 4.** *There is a polynomial-time algorithm that solves* WEIGHTED DIRECTED RING ROUTING *with load at most twice the optimum.*

**Sketch of the proof.** We solve an instance of Multicommodity Flow [1] obtained by relaxing the integrality constraints in the formulation of MIN-WDRR above. Let $\{x_1^*, \ldots x_n^*\}$ be the fractional optimal solution: the $h$th component $x_h^*$ is rounded to 0 if its value is $< 1/2$, to 1 otherwise. □

We now use Lemma 4 to obtain a polynomial-time approximation scheme (PTAS) for WDRR. Let $\overline{L}$ be the load obtained by applying the algorithm described above. In the following, $\epsilon$ denotes any fixed positive number. Following [14], a call is routed *long-way* if it uses the longer path to connect its end vertices (ties are broken arbitrarily), it is routed *short-way* otherwise; futhermore, a call $C_h$ is *heavy* if $l_h \ge \epsilon\overline{L}/3$, it is *light* otherwise. The number of heavy calls routed long way is bounded, as stated in the following lemma.

**Lemma 5.** *In any optimal solution there are at most* $12/\epsilon$ *heavy, long-way routed calls.*

Let $H \subseteq \mathcal{C}$ denote the set of heavy calls. For any set $S \subseteq H$, let $LP_S$ denote the following linear program:

$\min L(S) = \max\{\max_v A_v, \max_v B_v\}$

$$A_v = \sum_{h:[v,v+1]\subseteq[s_h,t_h]} l_h x_h + a_v, \quad B_v = \sum_{h:[v+1,v]\subseteq[t_h,s_h]} l_h(1-x_h) + b_v, \quad x_h \subseteq [0,1], 1 \le h \le n.$$

$a_v$ ($b_v$) denotes the load on clockwise (counter clockwise) directed arc $(v, v+1)$ ($(v+1, v)$) resulting from routing long-way calls belonging to $S$ and short-way calls belonging to $H - S$. Finally, $L^*(S)$ denotes the value of the optimal (fractional) solution of $LP^*(S)$. We now give the PTAS for MIN-WDRR:

**Algorithm WDRR-PAS**($\epsilon$)
**input:** $\langle \Sigma_m, \mathcal{C} \rangle$;
**output:** routing for $\langle \Sigma_m, \mathcal{C} \rangle$;
**begin** (1)
  **for** each $S \subseteq H : |S| \leq 12/\epsilon$ **do**
    **begin** (2)
      **for** each call $C \in H$ **do**
        **if** $C \in S$ **then** route $C$ long-way
        **else** route $C$ short-way;
      solve $LP_S$
    **end**(2);
  Let $y$ be the solution corresponding to $S'$ such that $L^*(S') = \min_S L^*(S)$;
  Apply a rounding procedure to $y$ to a feasible routing $\tilde{x}$ with load $\tilde{L}(S')$;
  Output $\tilde{x}$
**end**(1).

Lemma 6 below shows that, for every $S \subseteq H$, $|S| \leq 12/\epsilon$, we can round the solution of $LP_S$ in such a way that the value $\tilde{L}(S)$ of the integer solution obtained differs at most $\epsilon \overline{L}/2$ from $L^*(S)$. In order to prove the lemma, following [19] we first give the following definition: two calls $C_h = [s_h, t_h, l_h]$ and $C_j = [s_j, t_j, l_j]$ are *parallel* if the intervals $[s_h, t_h]$ and $[t_j, s_j]$ or the intervals $[t_h, s_h]$ and $[s_j, t_j]$ intersect at most at their endpoints. It may eventually happen that $s_h = s_j$ and $t_h = t_j$. Viewing an arc also as a chord, a demand is parallel to an arc when it can be routed through that arc, otherwise it is parallel to that arc's reverse. Finally, for any optimal solution $x^* = \{x_1^*, \ldots, x_n^*\}$ to $LP(S)$, call $C_h$ is said *split* with respect to $x^*$ if $0 < x_h^* < 1$.

**Lemma 6.** *For any $S \subseteq H$, $|S| \leq 12/\epsilon$, there is a polynomial time rounding procedure of the solution of $LP_S$, such that $\tilde{L}(S) - L^*(S) \leq \epsilon L/2$.*

*Proof.* Given $S \subseteq H$, $|S| \leq 12/\epsilon$, let $x^*$ be a solution to $LP_S$ with value $L^*(S)$. Following [19] we first obtain a new fractional solution $\hat{x} = \{\hat{x_1}, \ldots, \hat{x_n}\}$ such that no pair of parallel calls are both split and whose value $\hat{L}(S)$ is not larger than $L^*(S)$. Let us assume that there is a pair $C_i = [s_h, t_h, l_h]$ and $C_j = [s_j, t_j, l_j]$ of parallel calls, with $0 < x_h^*, x_j^* < 1$. We now reroute them in such a way that only one of them remains split. Two cases may arise:

1. $x_h^* + l_j x_j^*/l_h \leq 1$. In this case we set $\hat{x_h} = x_h^* + l_j x_j^*/l_h$ and $\hat{x_j} = 0$. Consider now a clockwise directed arc $(v, v+1)$ belonging to the interval $[s_h, t_h] \cap [s_j, t_j]$ and let $A_v^*, \hat{A}_v$ be its loads respectively before and after rerouting. It is easily verified that $\hat{A}_v = l_h x_h^* + l_j x_j^* = A_v^*$. If instead $(v+1, v)$ is any counter clockwise directed arc in $[t_h, s_h] \cap [t_j, s_j]$, it is easily seen that, if $B_v^*$ and $\hat{B}_v$ respectively denote its loads first and after rerouting, we have $\hat{B}_v = (1 - x_h^*)l_h + (1 - x_j^*)l_j = B_v^*$. If $s_h \neq s_j$ or $t_h \neq t_j$, it also holds that arcs not belonging to $[s_h, t_h] \cap [s_j, t_j]$ or $[t_h, s_h] \cap [t_j, s_j]$ have the same or reduced loads.

2. $x_h^* + l_j x_j^*/l_h > 1$. In this case we set $\hat{x_h} = 1$ and $\hat{x_j} = x_j^* + l_h/l_j(x_h^* - 1)$. The analysis of this case proceeds exactly as in the previous one. Again we have

$\hat{A}_v = A^*_v$ for any clockwise directed arc belonging to interval $[s_h, t_h] \cap [s_j, t_j]$ and for any counter clockwise directed arc belonging to interval $[t_h, s_h] \cap [t_j, s_j]$, while the load on any other arc, if any, doesn't increase or decrease. As before, a similar argument holds for counter clockwise directed arcs.

At the end of this procedure we have a new solution $\hat{x}$ whose load $\hat{L}(S)$ is at most $L^*(S)$ and such that no two split demands are parallel. Without loss of generality we may assume that calls in $\mathcal{C}$ are ordered in such a way that $\{C_1, \ldots, C_q\}$ is the set of split calls for solution $\hat{x}$. We are now left to route split calls. Given any clockwise (counter clockwise) directed arc $(v, v+1)$ $((v+1, v))$, let $F_v \subseteq \{C_1, \ldots, C_q\}$ be the set of all split calls that are parallel to $(v, v+1)$ $((v+1, v))$. Observe that calls in $F_v$ represent an interval in the ordered set $\{C_1, \ldots, C_q\}$; in the following, we shall denote $F_v$ as an interval $[i_v, j_v]$, where $i_v$ and $j_v$ respectively are the first and last index of calls in $\{C_1, \ldots, C_q\}$ that are parallel to $(v, v+1)$ (or $(v+1, v)$). Also observe that $F_v$ is exactly the set of all and only the calls whose rerouting can potentially increase the load on $(v, v+1)$ (or $(v+1, v)$). Let $\tilde{A}_v$ (respectively $\tilde{B}_v$) denote the load resulting on $(v, v+1)$ (respectively $(v+1, v)$) after routing calls that are split in $\hat{x}$ and let $\tilde{x} = \{\tilde{x}_1, \ldots, \tilde{x}_n\}$ be the corresponding integer solution. Finally, let $\tilde{L}(S) = \max\{\max_v \tilde{A}_v, \max_v \tilde{B}_v\})$. We have:

$$\tilde{A}_v = \hat{A}_v + \sum_{h \in [i_v, j_v]} l_h(\tilde{x}_h - \hat{x_h}),$$

$$\tilde{B}_v = \hat{B}_v + \sum_{h \notin [i_v, j_v]} l_h[(1 - \tilde{x}_h) - (1 - \hat{x_h})] = \hat{B}_v + \sum_{h \notin [i_v, j_v]} l_h(\hat{x_h} - \tilde{x}_h).$$

We are now ready to route calls that are split in $\hat{x}$. For $j = 1, \ldots, q$, $C_j$ is routed as follows:

$$\tilde{x}_j = \begin{cases} 1 \text{ if } -l_j\hat{x_j} + \sum_{h=1}^{j-1} l_i(\tilde{x}_h - \hat{x_i}) < -\frac{l_j}{2} \\ 0 \text{ otherwise} \end{cases}$$

As a consequence, if $\tilde{x}_j = 1$ then $\sum_{h=1}^{j} l_h(\tilde{x}_h - \hat{x_h}) < l_j/2$, while if $\tilde{x}_j = 0$ then $\sum_{h=1}^{j} l_h(\tilde{x}_h - \hat{x_h}) \geq -l_j/2$. In both cases $\sum_{h=1}^{j} l_h(\tilde{x}_h - \hat{x_h}) \in \left[-\frac{l_j}{2}, \frac{l_j}{2}\right)$ for any $j \in [i_v, j_v]$.

We now show that, for each clockwise directed arc $(v, v+1)$, $\tilde{A}_v - \hat{A}_v \leq (3/2)l_{max}$, where $l_{max} = \max_{h \in \{1,\ldots,q\}} l_h$. Since we are considering light calls, we have $l_{max} \leq (\epsilon L)/3$. This implies $\tilde{A}_v - \hat{A}_v \leq (\epsilon L)/2$. In particular, given any clockwise directed arc $(v, v+1)$, two cases may arise:

1. $i_v \leq j_v$. In this case we bound $\tilde{A}_v - \hat{A}_v = \sum_{h=i_v}^{j_v} l_h(\tilde{x}_h - \hat{x_h})$ as follows:

$$\tilde{A}_v - \hat{A}_v = \sum_{h=1}^{j_v} l_h(\tilde{x}_h - \hat{x_h}) - \sum_{h=1}^{i_v - 1} l_h(\tilde{x}_h - \hat{x_h}) \leq \frac{l_j}{2} - \left(-\frac{l_j}{2}\right) = l_j \leq l_{max}.$$

2. $i_v > j_v$. In this case we have:

$$\tilde{A}_v - \hat{A}_v = \sum_{h=1}^{q} l_h(\tilde{x}_h - \hat{x_h}) - \sum_{h=1}^{i_v - 1} l_h(\tilde{x}_h - \hat{x_h}) + \sum_{h=1}^{j_v} l_h(\tilde{x}_h - \hat{x_h}) < \frac{3}{2} l_j.$$

Since $\frac{3}{2}l_j \leq \frac{3}{2}l_{max}$, it follows that $\tilde{A}_v - \hat{A}_v \leq (3/2)l_{max}$. Since we are only considering light calls (heavy calls have already been routed) and recalling the definition of light call, $\tilde{A}_v - \hat{A}_v \leq (\epsilon L)/2$. In the same way we can prove that $\tilde{B}_v - \hat{B}_v$. Since $\tilde{L}(S) - L^*(S) \leq \max_v\{\max\{A_v - \hat{A}_v, B_v - \hat{B}_v\}\}$ the thesis follows. □

This is sufficient to prove the main result of this section. In fact, if $S_{OPT}$ denotes the set of heavy calls that are routed long-way in the optimal integer solution, we have $\tilde{L}(S) \leq \epsilon\overline{L}/2 + L^*(S_{OPT})$ where, by lemma 4, $\epsilon\overline{L} \leq 2L^*(S_{OPT})$. This implies that $\tilde{L}(S) \leq (1+\epsilon)L^*(S_{OPT})$; since $L^*(S_{OPT})$ is a lower bound to the optimal integer solution the following theorem holds

**Theorem 4.** *The solution provided by algorithm* **WDRR-PAS**$(\epsilon)$ *has value at most* $(1+\epsilon)OPT$. *For any fixed* $\epsilon > 0$, *the time complexity of the algorithm is polynomial.*

### 4.2 An Approximate Algorithm for RING-MIN-CS

When $\epsilon > 0$, Algorithm WDRR-PAS($\epsilon$) finds a $(1+\epsilon)$-approximate algorithm for routing with minimum load. We now turn to the scheduling phase. In the following $T^*$ denotes the makespan of an optimal solution to MIN-CS.

A simple idea is as follows: *i)* solve MIN-WDRR within $(1+\epsilon/2)$ the optimum, *ii)* cut the ring at any link, say $e$, *iii)* solve two instances of MIN-CHAIN-CS with calls that do not use link $e$ (in any direction), *iv)* when the last call scheduled in the previous step is completed schedule calls that use link $e$ using a $(1+\epsilon/2)$ approximate partitioning algorithm.

It seems reasonable to prove that the above simple algorithm gives an approximation ratio of $6+\epsilon$. However this is not necessarily true because the optimum scheduling algorithm might use a completely different routing of calls (even one that is not $\epsilon$ close to the routing with minimum load). It is possible to prove the following weaker result, whose proof is omitted here. In the full paper we will give a counterexample showing that, using the routing obtained in the previous subsection one is unlikely to prove anything better than $11+\epsilon$.

**Theorem 5.** *For any fixed* $\epsilon > 0$, *there exists a polynomial-time algorithm that finds a routing and a schedule for* RING-MIN-CS *having makespan* $\leq (12+\epsilon)T^*$.

We conclude this section by observing that the same result holds also in the case of undirected rings. In this case we will use the polynomial time approximation scheme proposed in [14] to compute the routing of calls. Details will be given in the full paper.

## 5 Conclusions

In this paper we have considered the Call Scheduling problem in all-optical, networks. We have proposed approximation algorithms with constant approximation ratio for both chain and ring networks. As a side result, we have also

proposed a polynomial-time approximation scheme for the problem of routing weighted calls in a ring network with the aim of minimizing the maximum load, a problem which has itself a practical relevance in SONET networks [18,14,19].

It would be interesting to consider different network topologies and to consider the problem whith release dates and different objective functions. Also the on-line version of the problems deserves future investigation.

## References


1. R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network flows.* Prentice-Hall, 1993. 208
2. B. Awerbuch, Y. Azar, A. Fiat, S. Leonardi, and A. Rosen. On-line competitive algorithms for call admission in optical networks. In *Proc. of the 4th Annual European Symposium on Algorithms*, 1996. 202
3. Y. Bartal, A. Fiat, and S. Leonardi. Lower bounds for on-line graph problems with application to on-line circuit and optical-routing. In *Proc. of the 28th Annual Symposium on the Theory of Computing*, 1996. 202, 202
4. Y. Bartal and S. Leonardi. On-line routing in all-optical networks. In *Proc. of the 24th International Colloquium on Automata, Languages and Programming*, 1997. 202
5. L. Becchetti. *Efficient Resource Management in High Bandwidth Networks.* PhD thesis, Dipartimento di Informatica e Sistemistica, University of Rome "La Sapienza", 1998. 203
6. C. Brackett. Dense Wavelength Division Multiplexing Networks: Principles and Applications. *IEEE Journal Selected Areas in Comm.*, 8, 1990. 201
7. T. Erlebach and K. Jansen. Scheduling Virtual Connections in Fast Networks. In *Proc. of the 4th Parallel Systems and Algorithms Workshop PASA '96*, 1996. 202
8. T. Erlebach and K. Jansen. Call Scheduling in Trees, Rings and Meshes. In Proc. of the 30th Hawaii International Conference on System Sciences, 1997. 202
9. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, San Francisco, 1979. 204
10. J. Gergov. Approximation algorithms for dynamic storage allocation. In *Proc. of the 4th Annual European Symposium on Algorithms*, 1996. 204, 205, 205, 207
11. J. Gergov. Algorithms for Compile-Time Memory Optimization. Private communication, 1998. 202, 204, 204
12. R. L. Graham. Bounds for multiprocessing timing anomalies. *SIAM Journal of Applied Math.* , 17, 1969. 206
13. P. E. Green. *Fiber-optic Communication Networks.* Prentice-Hall, 1992. 201, 201, 201
14. S. Khanna. A Polynomial Time Approximation Scheme for the SONET Ring Loading Problem. *Bell Labs Tech. J.*, Spring, 1997. 202, 203, 208, 211, 212
15. H. A. Kierstead and W. T. Trotter. An extremal problem in recursive combinatorics. *Congressus Numerantium*, 33, 1981. 202
16. T. Leighton, B. Maggs, and S.Rao. Packet routing and jobshop scheduling in O(congestion+dilation) steps. *Combinatorica*, 14, 1994. 202
17. P. Raghavan and E. Upfal. Efficient Routing in All-Optical Networks. In *Proc. of the 26th Annual Symposium on the Theory of Computing*, 1994. 201, 202
18. A. Schrijver, P. Seymour, and P. Winkler. The Ring Loading Problem. *SIAM J. on Discrete Math.*, 11, 1998. 202, 202, 202, 203, 212

19. G. Wilfong and P. Winkler. Ring Routing and Wavelength Translation. In *Proc. of the European Symposium on Algorithms*, pages 333–341, 1998. 202, 203, 208, 209, 209, 212