
**Software Engineering Department**

**Ort Braude College**

**Course 61998: Extended Project in Software Engineering**

# Capstone Project Phase A

## Project Number: 23-1-R-19

## Online minimization of switching cost in optical networks

**Students:**

***Ahmad Aamar - 206713034***

**Email:** ahmad.aamar1410@gmail.com

***Mohamed abu assad - 316449495***

**Email:** m7mad114@gmail.com

**Supervisor:**

**Shmuel Zaks**

**Email:** shmuel.Zaks@braude.ac.il

**Commented [ma1]:** קודם כל שנינו שמנו את המייל הנכון.

**Commented [ma2]:** מספרנו את העמודים

# Contents


1. **Introduction** ..... 4

2. **Background and related work** ..... 5

   Introduction to Online algorithms ..... 5
   Introduction to Optical networks ..... 7
   Definitions ..... 8

3. **Our Contribution**:..... 13

4. **Expected achievements** ..... 14

5. **Research process** ..... 14

   The algorithm ..... 15

   Upper bounds..... 19
      General topology..... 19
      Ring topology ..... 23
      Path topology ..... 23

   Lower bounds..... 25
      General topology..... 25
      Path topology ..... 28
      Ring topology ..... 31

6. **Evaluation plan** ..... 33

7. **Summary** ..... 33

**References** ..... 34


**Commented [ma3]:** קשרנו כל נושא למקומו המתאים בספר

**ABSTRACT**

Optical networks rely on optical wavelength switching devices, known as ADMs, to route light paths through the network. The efficient management of these ADMs is crucial for the performance and cost-effectiveness of the network. In this study, we address the problem of minimizing the number of ADMs used in optical networks that shown in [1]. Previous research on this problem has focused on offline scenarios, where all the light paths (i.e., the routes that light signals travel through the network) are known in advance. However, in real-world situations, light path requests arrive at the network dynamically and must be assigned wavelengths in a way that minimizes the cost of switching.

In this study, we present a deterministic online algorithm aimed at minimizing the number of Add-Drop Multiplexers (ADMs) in optical networks, as discussed in [1]. We proceed to demonstrate that the algorithm achieves a competitive ratio of 7/4, which quantifies its performance relative to the optimal offline solution. Furthermore, we establish that this competitive ratio represents the best achievable performance for the general case, indicating that no other online algorithm can achieve a better competitive ratio for a ring topology network. Additionally, we illustrate that the proposed algorithm achieves a competitive ratio of 3/2 for path topology networks, which is optimal for this specific network type.

The analyses of the upper bounds and lower bounds for the various topologies are established using a variety of proof techniques, which are of interest in their own right and may be useful for future research on this topic. In summary, our work makes significant contributions to the theory of optimal on-line colorings for minimizing the number of ADMs in optical networks and has practical implications for the design and operation of real-world optical networks.

## 1. INTRODUCTION

The field of communication networks is currently facing an unprecedented increase in data traffic, which has led to the adoption of a technology called optical wavelength-division multiplexing (WDM) to transport large amounts of data between sites. In optical networks, a lightpath refers to a path between two nodes that is assigned a wavelength. Each lightpath is composed of two elements called Add-Drop multiplexers (ADMs), and the total number of ADMs used in a network is considered to be the total cost of the network.

In this project, we aim to minimize the number of ADMs used in a given WDM network by assigning a wavelength to each lightpath in the network, as discussed in [1]. The problem can be approached in two different ways: offline and online. In an offline case, the entire set of lightpaths is given to the algorithm before it begins assigning wavelengths. In contrast, in an online case, lightpaths arrive in the network one at a time, and the algorithm assigns wavelengths to them without knowledge of future lightpaths that will arrive.

To minimize the total cost of the network, we take into account the constraint that two lightpaths with the same wavelength that share an endpoint, but do not share an edge, can use the same ADM and be considered as one longer path. However, it should be noted that an ADM can be shared by at most two lightpaths, as it is designed to be used in ring and path networks, where the degree of every node is at most two.

In the next section, we will provide background information on optical networks and online algorithms, as well as explain our expected achievements for this project. In Section 5, we will detail the research process and the algorithm that has been discussed in [1], including analyses of upper and lower bounds of the algorithm in different topologies such as path, ring, and general topologies. In Section 6, we will outline our evaluation plan, and in Section 7, we will conclude the project.

**Commented [ma4]:** * עשינו התאמה לבין מה שכתוב כאן ולמה נמצא בספר

# 2.Background & related work:

## 2.1 Online Algorithms:

Online algorithms are a type of algorithm that processes data as it arrives, rather than waiting for all the data to be collected before processing it. These algorithms are designed to handle data streams and make decisions or take actions based on the data as it arrives. Online algorithms are widely used in various fields, such as real-time systems, machine learning, and network management.

One of the main characteristics of online algorithms is that they have to make decisions without the knowledge of future inputs. This makes online algorithms different from offline algorithms, which have access to the entire input dataset before making any decisions. As a result, online algorithms need to be designed to handle uncertainty and make decisions based on limited information. In addition, the lack of knowledge of the entire input in an online algorithm can lead to decisions that are not optimal.

In network management, online algorithms are used to dynamically allocate resources, such as bandwidth and routing, in response to changing network conditions. These algorithms are designed to make decisions in real-time and have to be efficient, scalable and robust enough to handle the large number of requests that arrive in a network.

To evaluate the performance of an online algorithm, a methodology called competitive analysis is commonly employed. Competitive analysis provides a framework for measuring how well an online algorithm approximates the optimal solution. An online algorithm A is said to be c-competitive if there exist positive constants α and c, such that for any finite input sequence I, the cost incurred by A (denoted as A(I)) is no more than c times the cost of the optimal offline algorithm (denoted as OPT(I)) plus α. When α is less than or equal to zero, the algorithm A is considered strictly c-competitive.

**Commented [ma5]:** פרטנו יותר על אלגוריתמים מכוונים, דברנו יותר על שלב הניתוח ו נתתנו כמה דוגמאות.

In simpler terms, the competitive ratio of an online algorithm is the ratio between the cost incurred by the algorithm and the cost of the optimal offline algorithm. A lower competitive ratio indicates a better approximation of the optimal solution. The concept of competitiveness provides a quantitative measure of how well an online algorithm performs in comparison to an offline algorithm.

To illustrate the notion of competitive ratio, let's delve into the "rent-or-buy" problem mentioned in [3]. Imagine going on a ski vacation where the number of ski days is unknown. Renting skis costs $10 per day, while buying skis costs a fixed price of $100. The objective is to minimize the total cost of skiing. If the number of ski days were known in advance, the decision between renting and buying would be straightforward. If the number of days exceeds 10, it is more cost-effective to

**Commented [ma6]:** ** השתמשנו ברפרנסים

purchase skis on the first day. However, if the number of days is less than or equal to 10, renting skis would be the better option.

In the context of competitive analysis, the worst-case scenario is considered. Suppose you decide to buy the skis and end up skiing only once during your vacation. In this case, you would pay $100, whereas the optimal solution would be to rent skis for $10. Hence, the competitive ratio for this scenario is 100/10 = 10. However, by employing a different strategy, such as renting skis until realizing the need to buy, it is possible to achieve a competitive ratio of ≤ 2.

Another illustrative example of online algorithms is the "list accessing problem" discussed in [2]. In this problem, you have a filing cabinet containing l labeled and unsorted files. Requests to access specific files arrive one by one, and the goal is to minimize the cost of accessing these files. Each access operation incurs a cost equal to the position of the accessed file in the list. Additionally, if a file is not found, it must be returned to the cabinet before the next request arrives. Reorganizing the list incurs a cost equal to the current number of files plus one.

** :Commented [ma7]

The lower bound for this problem is established through a proof. It is shown that for any deterministic online algorithm, the competitive ratio is at least 2 - 2/(l+1). This implies that no deterministic online algorithm can achieve a competitive ratio better than this lower bound. Thus, the competitive analysis provides insights into the inherent limitations of online algorithms for certain problems.

Online algorithms find applications in various domains, including data structures and graph theory. For instance, in the implementation of an Abstract Data Type (ADT) called a

priority queue, an online algorithm may be used to handle incoming insertions and deletions in an efficient manner. Similarly, online algorithms are employed in graph coloring problems, where the goal is to assign colors to the vertices of a graph subject to certain constraints. The ability of online algorithms to make decisions based on partial information makes them well-suited for these types of problems.

In conclusion, online algorithms are algorithms that process input data incrementally as it becomes available, without knowledge of the entire input in advance. Competitive analysis is a valuable tool for assessing the performance of online algorithms by comparing their results to an optimal offline solution. While online algorithms may not always achieve the same level of optimality as offline algorithms,

they play a crucial role in real-time decision-making scenarios and provide efficient solutions for a wide range of computational problems.

In this project, we will be focusing on online algorithm that are used to assign wavelengths to lightpaths in optical networks, as discussed in [1], with the goal of minimizing the total cost of the network by minimizing the number of ADMs used.

## 2.2 Optical Networks:

Optical networks are communication networks that use light waves to transmit data. These networks are built using optical fibers, which are thin strands of glass or plastic that can carry light over long distances. The light waves used in optical networks are at different frequencies, known as wavelengths, and this feature is used to multiplex multiple channels of data onto a single optical fiber, a process known as wavelength-division multiplexing (WDM).

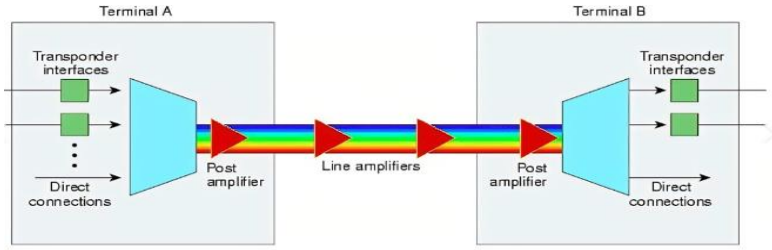


Fig.1: wavelength-division multiplexing (WDM). (See more in [4])

** :Commented [ma8]

Optical networks have several advantages over traditional copper-based networks, such as increased bandwidth, longer transmission distances, and greater resistance to electromagnetic interference. These advantages have led to the widespread adoption of optical networks in various fields, including telecommunications, data centers, and long-distance data transmission.

In optical networks, data is transmitted in the form of optical signals, which are generated by a device called a transmitter and received by a device called a receiver. These signals are transmitted through optical fibers, which are connected to various network elements such as amplifiers, multiplexers, and switches. These network elements are used to control and manage the flow of data within the network.

One of the key elements of optical networks is the Add-Drop Multiplexer (ADM), which is used to add or drop specific wavelengths of light to or from a specific optical fiber. ADMs are used to create lightpaths, which are paths between two nodes in the network that are assigned a specific wavelength. The total number of ADMs used in a network is considered to be the total cost of the network, and minimizing this cost is an important goal in the design and management of optical networks.

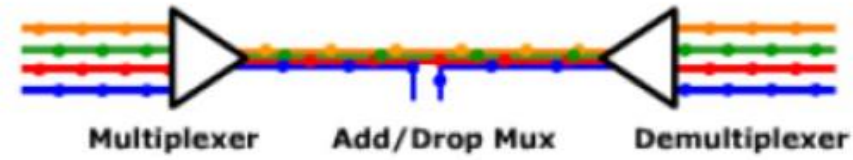


Fig.2: Add-Drop Multiplexer (ADM).(see more details in [5])

** :Commented [ma9]

**multiplexer:** A multiplexer (MUX) is a device that combines multiple signals into one signal for transmission over a shared medium. It allows multiple data streams to be transmitted over a single communication channel by switching between the inputs at a high rate

**Demultiplexer:** A demultiplexer (DEMUX) is a device that separates a single signal into multiple signals. It is the inverse of a multiplexer and is used to extract individual data streams from a multiplexed signal

## 2.3 Definitions:

We introduce the definitions used when presenting and analyzing ONLINE-MINADM, an online algorithm for minimizing the number of ADMs in optical networks.

**Definition 1:** An instance of the problem, denoted as α = (G, P), is composed of an undirected graph G = (V, E) and a set of simple paths P. In the case of an online instance, the graph G is fixed and known in advance, while the set P of paths is presented one at a time. The input is represented by P=$\{p_1, p_2 \dots, p_N\}$ with $p_i$ being the i-th path and $P_i = \{p_j \in P \mid j \leq i\}$ being the first i paths in the input.

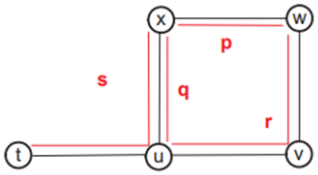


Fig.3: sample input. (See more in [1])

** :Commented [ma10]

**Definition 2:** Two paths p, p' in P are considered conflicting or overlapping if they share at least one edge. This is represented as $p \asymp p'$. The graph that represents this relation of overlapping is called the conflict graph of the pair (G, P).

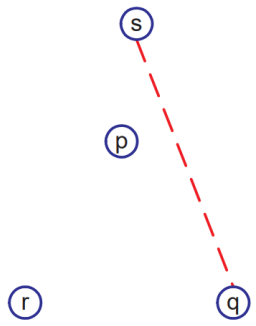


Fig.4: conflict graph. (See more in [1])

** :Commented [ma11]

**Definition 3:** A proper coloring or wavelength assignment of P is a function w: P -> N, such that for any two conflicting paths p, p' it holds that $w(p) \neq w(p')$. It can be observed that w is a proper coloring if and only if for any color c in N, the set of paths that are assigned color c, represented as $w^{-1}(c)$, is an independent set in the conflict graph.

**Definition 4:** A valid chain in an instance $\alpha = (G, P)$ is a path that is composed of multiple distinct paths $p_{i0}$, $p_{i1}$, ..., $p_{i\,k+1}$ from P, without traversing any edge twice. Similarly, a valid cycle is a cycle formed by the concatenation of distinct paths from P that do not go over the same edge twice. It's important to note that the paths included in a valid chain or cycle form an independent set in the conflict graph.

**Definition 5:** A solution of an instance $\alpha = (G, P)$ is a collection of valid chains and cycles that covers all the paths in P. Each path in P must be included in exactly one chain or cycle, and no two chains or cycles share the same set of nodes in the conflict graph G.

**Definition 6:** The shareability graph of an instance $\alpha = (G, P)$, is a graph $G_\alpha = (P, E_\alpha)$ where P is the set of paths, $E\alpha$ is the set of edges. There's an edge e = (p, q) labeled u in $E\alpha$ if p and q are non-conflicting paths and u is a common endpoint of p and q in G.

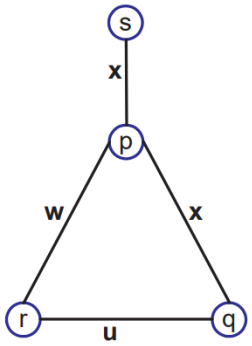


Fig.5: shareability graph. (See more in [11])

** :Commented [ma12]

The example provided illustrates the shareability graph of an instance α = (G, P) as shown in **Figure 3**. The shareability graph, denoted as $G_\alpha = (V_\alpha, E_\alpha)$ is represented in **Figure 5**. In this example, P = {p, q, r, s} is the set of nodes of Gα. The edges and their labels in $E_\alpha$ are defined as {(q, r, u), (p, r, w), (p, q, x), (p, s, x)} because p and q can be joined at their common endpoint x, and so on. It's worth noting that, for example, (q, s, x) is not in Eα as they share a common endpoint x but they cannot be concatenated because they have a common edge (x, u). The corresponding conflict graph is depicted on the right side of **Figure 2**, it has the same set of nodes and one edge (q, s) as paths q and s are conflicting because they share a common edge (u, x). It's worth noting that the edges in the conflict graph are not in $E_\alpha$ as per the definition. Also, for any node v in $G_\alpha$, the number of labels of the edges adjacent to v is at most two.

**Definition 7:** A valid chain or cycle in $G_\alpha$ is a simple path or cycle in $G_\alpha$ that has edges with instinct labels and a node set that can be colored using one color (or forms an independent set in the conflict graph G). These valid chains and cycles in $G_\alpha$ correspond to valid chains and cycles in the original instance α. For example, a chain such as p, s that is formed by concatenating paths p and s in G, corresponds to a simple path p, s in $G_\alpha$. Similarly, a cycle such as p, q, r that is formed by concatenating three paths in G, corresponds to the cycle p, q, r in $G_\alpha$, as long as no two consecutive labels in the cycle are equal. However, paths such as q, p, s cannot be concatenated to form a chain, because it would require connecting p to both q and s at node x. This would correspond to a path q, p, s in $G_\alpha$, which is not valid because the edges (q, p) and (p, s) have the same label, namely x.

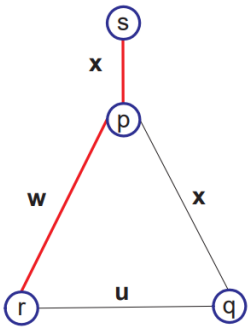


**Fig.6**: sharing graph of the solution S = {(s, p, r), (q)} for the input in Figure 3. The thick lines are the edges in the sharing graph. (See more in [11])

** :Commented [ma13]

**Definition 8:** The sharing graph of a solution S of an instance α = (G, P) is a subgraph of $G_\alpha$ called $G_{\alpha, S} = (P, E_S)$ that is formed by connecting lightpaths in P when they are consecutive in a chain or cycle in the solution S, and their common endpoint is u ∈ V. The edges in $E_S$ are labeled with the common endpoint u. This subgraph is usually referred to as $G_S$. The degree of a node p in $G_S$ is denoted by d(p). such that, d(p) is the number of edges incident to p. It represents the number of connections a node has in the graph.

The given solution, S, is a set of two chains, (s, p, r) and (q), that satisfies the distinct labelling condition and independent set condition. The distinct labelling condition is satisfied vacuously for chains of size at most two, meaning that it is not necessary to have distinct labels for these chains. The independent set condition is satisfied because no edge of the conflict graph, can be an edge of the sharing graph Gα. The sharing graph of this solution is shown in **Fig 6** and it describes a sharing graph of solution for the input that given in **Fig 3.**

**Definition 9:**

The partitioning of the set of lightpaths P into disjoint subsets based on the degree of the corresponding node in $G_S$ can be useful for analyzing the structure of the solution. Each subset will contain the lightpaths corresponding to nodes with the same degree in $G_S$. This partitioning can help in understanding the distribution of lightpaths and the connectivity of the nodes in the sharing graph. It can be used to identify any patterns or anomalies in the solution that may affect its performance or robustness. Therefore, for any solution S, we partition the set of lightpaths P into 3 disjoint subsets such that:

For every $i \in \{0,1,2\}$, $D_i$ of $S$ is a set of paths where d(p)= i.

For the example in **Fig 6,** the disjoint subsets are:

$D_0$={q}, $D_1$= {s, r}, $D_2$={p}.

**Definition 10:**

Similar to definition 9, we define for every $i \in \{0,1,2\}$ $d_i(S)$, such that:

$d_i(S)= |D_i(s)|$.

For the example in **Fig 6:**

$d_0(S)= |D_0(s)|=1$, $d_1(S)= |D_1(s)|=2$, $d_2(S)= |D_2(s)|=1$.

**Definition 11:**

An edge (p,q) in $E_S$ represents a concatenation of two paths with the same color at their common endpoint u. This means that the two lightpaths p and q can share an ADM operating at node u, thus saving one ADM. Therefore, every edge in ES corresponds to a saving of one ADM. When no ADMs are shared, each path needs two ADMs, resulting in a total of 2N ADMs. The presence of edges in ES implies that some ADMs are shared and thus less ADMs are required overall, reducing the cost of the network. Base on that, we define **cost(S)** such that:

$$cost(S)= 2 |P| - |E_S|= 2N - |E_S|.$$

The objective of the problem is to find a solution S such that the cost of S is minimum, which is equivalent to finding a solution such that the number of edges in $E_S$, $| E_S |$, is maximum. This is because each edge in $E_S$ corresponds to a saving of one ADM, and therefore a maximum number of edges in $E_S$ represents a minimum cost. In other words, the goal is to find a solution S where as many edges as possible can be shared between paths at a common endpoint, thus reducing the number of ADMs required and minimizing the cost of the network.

**Definition 12:**

Given a solution S, the degree of every node p in P, d(p), is at most 2. This means that each node can be connected to at most 2 other nodes in $G_S$. As a result, the connected components of $G_S$ are either paths or cycles. An isolated node is a special case of a path. Let $P_S$ be the set of the connected components of $G_S$ that are paths. It is clear that the number of edges in $E_S$, $| E_S |$, is equal to the number of nodes in P, N, minus the number of connected components in $P_S$ which are paths. This is because each edge in $E_S$ represents a shared endpoint between two paths, and every path in $P_S$ represents an endpoint that is not shared with any other path. Therefore:

$$cost(S) = 2N - | E_S | = N + | P_S |.$$

**Definition 13:**

**Let $S^*$** be a solution with minimum cost. For any solution S we define:

$$\varepsilon(\mathbf{S}) = \frac{\mathbf{d0}(\mathbf{S}) - \mathbf{d2}(\mathbf{S}) - \mathbf{2}\,|\mathbf{PS}*|}{\mathbf{N}}$$

**Lemma 1.1** For any solution S:

$$\text{cost}(S) = \text{cost}(S^*) + {}^{1}/_{2}\, N\,(1 + \varepsilon(S))$$

**Proof:**

From definition 12 we can fine that $2N - |E_{S^*}| = N + |P_{S^*}|$. Thus $N = |E_{S^*}| + |P_{S^*}|$.

So Clearly $|E_{S^*}| = N - |P_{S^*}|$. On the other hand, the sum of the degrees of the nodes in GS is equal to $2|E_S|$. This is because each edge in $G_S$ is incident to two nodes, and the degree of a node is the number of edges incident to it. So:

$$2\,|E_S| = d_1(S) + 2d_2(S) = N - d_0(S) + d_2(S)$$

On conclusion:

$$\text{cost}(S) - \text{cost}(S^*) = |E_{S^*}| - |E_S| = N - |P_{S^*}| - \frac{N - d0(S) + d2(S)}{2}$$

$$= \frac{\mathbf{N}}{\mathbf{2}} + \frac{\mathbf{d0}(\mathbf{S}) - \mathbf{d2}(\mathbf{S}) - \mathbf{2}\,|\mathbf{PS}*|}{\mathbf{2}} = \frac{\mathbf{N}}{\mathbf{2}}\left(\mathbf{1} + \frac{\mathbf{d0}(\mathbf{S}) - \mathbf{d2}(\mathbf{S}) - \mathbf{2}\,|\mathbf{PS}*|}{\mathbf{N}}\right)$$

## 3. Our Contribution:

* :Commented [ma14]

We will demonstrate the algorithm shown in [1] that performs well in any network topology, with a competitive ratio of ${}^{7}/_{4}$. We will elaborate the proves shown in [1], that no deterministic online algorithm can achieve a better competitive ratio, even when the topology is a ring. In path topologies, this algorithm has a competitive ratio of ${}^{3}/_{2}$ and this is the lowest possible ratio for online algorithms in this type of topology. However, the lower bound for the competitive ratio on a ring topology does not apply when the ring is of a limited size. The analyses for the upper bounds and lower bounds for these algorithms use a variety of proof techniques that are valuable on their own and may be useful in future research on this topic. In the following sections, we will describe the problem and some initial results, present the algorithm and its competitive analysis, provide lower bounds and upper bounds for the competitive ratio on various topologies (in section 5).

## 4. Expected Achievements:

- Develop and implement a deterministic online algorithm for minimizing the number of ADMs in optical networks.
- Analyze the performance of the proposed algorithm and compare it to the optimal offline solution.
- Determine the competitive ratio of the proposed algorithm for various network topologies, including ring and path topologies.
- Prove upper and lower bounds for the competitive ratio of the proposed algorithm for these topologies.
- Evaluate the effectiveness and efficiency of the proposed algorithm through simulations and/or experiments.
- Contribute to the theoretical understanding of optimal on-line colorings for minimizing the number of ADMs in optical networks.

## 5. Research Process:

In this project, we address the problem of minimizing the cost of an optical network by minimizing the number of ADMs (optical wavelength switching devices) required. Previous studies on this problem have focused on the offline scenario, where all the light paths are known in advance. However, in real-world situations, light path requests arrive dynamically and must be assigned wavelengths in a way that minimizes the cost. To tackle this challenge, we illustrate the online algorithm that shown in [1], as in real-life the input of a network arrives dynamically and the algorithm deals with each input without knowing what will come next. In part A of the project, we began by researching new areas such as optical networks and online algorithms, and studied the algorithm presented in next section, adding remarks, examples, and detailed proofs as an addition to the paper. In part B of the project, we will develop a software to visualize the algorithm and test its average performance. While the analysis of the algorithm will be discussed later, based on the competitive ratio, which is an analysis of the worst case, this project aims to also analyze the average case.

## 5.1 Algorithm ONLINE-MINADM:

The ONLINE-MINADM algorithm is an online strategy for managing lightpaths in a network that has a $^{7}/_{4}$ -competitive ratio. This means that in any sequence of lightpaths, the number of ADMs used by the algorithm is at most $^{7}/_{4}$ times the number of ADMs used by the optimal offline strategy.

When lightpaths are added one-by-one to a network, the algorithm uses a simple coloring procedure. A new lightpath with endpoints u and v first checks for available ADMs at its endpoints. An ADM is considered free if it is only serving one lightpath that ends at that endpoint and does not share an edge with the new lightpath. If there are two ADMs of the same color, the algorithm will first try to connect them to form a cycle with existing lightpaths, and if that is not possible, a path is formed. If there are free ADMs available at one or both endpoints, the algorithm will try to connect to them. If there are no free ADMs, the new lightpath will be assigned a new color.

When trying to assign a color to a lightpath, pi, a color λ is considered feasible if it does not overlap with any other lightpaths that are already assigned the same color. In other words, if it is possible to assign the color λ to pi and still have a proper coloring for the set of lightpaths $P_i$.

**Main steps of the algorithm:**

When a lightpath $p_i$ with endpoints $u_i$ and $v_i$ arrives the algorithm checks the following conditions:

- If there is a chain of lightpaths with color λ that have endpoints $u_i$ and vi and color λ is feasible for lightpath $p_i$, then assign color λ to $p_i$. So that's mean that $w(p_i)= \lambda$.

- If a chain of lightpaths with color λ exists that has one endpoint from { $u_i$, $v_i$ } and color λ is feasible for lightpath $p_i$, then assign color λ to $p_i$.

- If no suitable color can be found, assign a new unused color λ′ to lightpath $p_i$.

It is important to note that when the algorithm resorts to using an unused color in the last clause, it ensures that no two chains will have the same color. Therefore, in the first clause, the algorithm will always form a cycle.

Consider a scenario where the lightpaths in Figure 3 are presented in the order of p, q, r, s:

The algorithm will assign two ADMs, one in node x and one in node w because p is the first path to arrive so there is no ADMs, so the algorithm will assign p to a new color so $w(p)= \lambda_1$. Then when q arrives the algorithm will that there is a free ADM that it can assign to it in x and assign to new ADM in u so $w(q)=w(p)= \lambda_1$. When r arrives the algorithm will close the circle because in the both of endpoints of r there is a free ADM with the same color, so $w(p)=w(q)=w(r)= \lambda_1$. When s arrives, the algorithm will assign two ADMs, one in node x and one in node t because there is no ADMs in the endpoints of s.

In conclusion, the solution S that given by ONLINE-MINADM algorithm is

S={(p,q,r),s} , and the cost=$N^2$-|$E_s$|=8-3=5 ADMs and that is the optimal for this example while if the order was s, p, r, q the cost will be 6 ADMs.

**Additional example:**

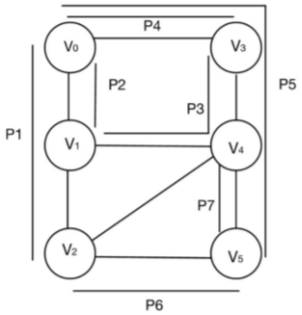


Fig.7. the input.

The input is a graph g={V,E} such that V={$v_0$,$v_1$,$v_2$,$v_3$,$v_4$,$v_5$} , E={ $e_0$=( $v_0$,$v_1$),

$e_1$= ($v_1$, $v_2$), $e_2$= ($v_3$, $v_4$), $e_3$= ($v_4$, $v_5$), $e_4$= ($v_0$, $v_3$), $e_5$= ($v_1$, $v_4$), $e_6$= ($v_2$, $v_4$), $e_7$= ($v_2$, $v_5$)} and the set of light paths P= {$p_1$= ($e_0$, $e_1$), $p_2$= ($e_0$), $p_3$= ($e_2$, $e_5$), $p_4$= ($e_4$), $p_5$= ($e_4$, $e_2$, $e_3$), $p_6$= ($e_7$), $p_7$=($e_3$)}, like in **Fig 7**.

For sequence of paths is: $p_1$, $p_5$, $p_6$, $p_2$, $p_4$, $p_3$, $p_7$.

when $p_1$ the algorithm assigns two ADMs in its endpoints ($v_0$, $v_2$) because it is the first path, so the algorithm defines new color λ=red, such that w($p_1$)=red, the algorithm will assign w($p_1$)=w($p_5$)= w($p_6$)=red, because when $p_5$ arrives he will find that there is an free ADM in node $v_0$ the color red is feasible for it, and when $p_6$ arrives it will find that there is one free ADM in $v_2$ and one free ADM in $v_5$ with the same color and the color red is feasible for it so the path $p_6$ will close the circle and save 2 ADMs, similar to that w($p_1$)=w($p_4$)=w($p_3$)= blue. finally, when $p_7$ arrives the algorithm well detect that there is no free ADMs for the path $p_7$ in its endpoints so it will define a new color λ=green such that w($p_7$) =green like in **Fig 8**.

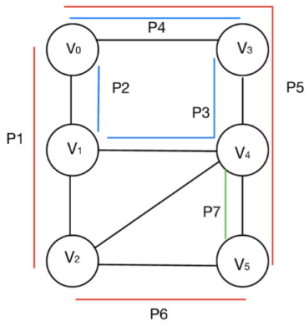

Fig.8. optimal solution

The solution S= {($p_1$, $p_5$, $p_6$),( $p_2$, $p_4$, $p_3$),$p_7$} and this is the optimal.

According to the Shareability graph for optimal solution in **Fig 9,** the optimal solution save $|E_s^*|$=6 ADMs, so:

Cost($S^*$) =2*|P|-$|E_s^*|$=14-6=8 ADMs.

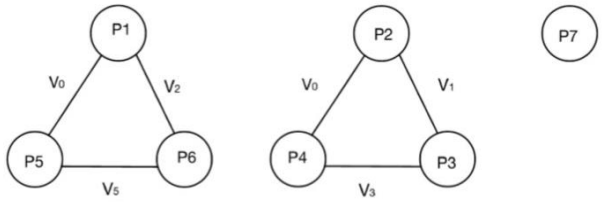


Fig.9. Shareability graph for optimal solution

For sequence of paths: $p_4$,$p_1$,$p_6$,$p_7$,$p_5$,$p_2$,$p_3$

The algorithm will assign new color for the first path $p_4$ because the network is empty so w($p_4$) = λ=blue, afterwards the algorithm will find that there is a free ADM for in node $v_0$ for $p_1$ so w($p_4$)= w($p_1$) =blue and similar to this w($p_6$)=blue and w($p_7$)=blue.

For $p_5$ the algorithm will define new color red and for $p_2$ and $p_3$ algorithm will define the same color λ=green , so w($p_7$)=blue, w($p_2$)= w($p_3$)=green. Like in **Fig 10.**

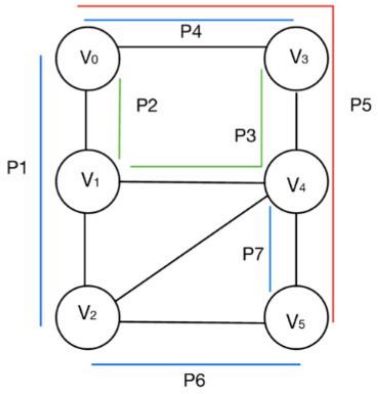


Fig.10. Online-MINADM solution

The solution S= {( $p_4$,$p_1$,$p_6$,$p_7$),( $p_2$,$p_3$),$p_5$}

According to the Shareability graph for solution in **Fig 11,** the solution saves $|E_s^*|$=6 ADMs, so:

Cost($S^*$) =2*|P|-|$E_S^*$|=14-4=10 ADMs.

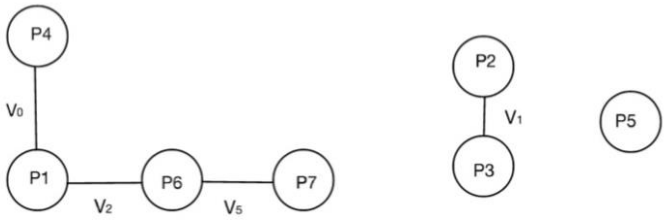


Fig.11. Shareability graph for Online-MINADM solution

The Online-MINADM algorithm is guaranteed to be correct because it assigns a proper coloring for the set of lightpaths Pi. If a lightpath pi is colored by the first two cases, it is checked for feasibility before being assigned a color. If it is not colored by the first two cases, a new unused color is assigned, thus ensuring no other lightpath conflicting with pi will have the same color. In the following sections, we will discuss the prove of this theorem.

The ONLINE-MINADM algorithm has been proven to be the most efficient for both general and ring network configurations, with a competitive ratio of 7/4. It is also the optimal solution for path network topologies, with a competitive ratio of 3/2. In the following sections, we will present both upper and lower limits and demonstrate that the ONLINE-MINADM algorithm is the best option for general, ring, and path network configurations.

## 5.2 Upper Bounds:

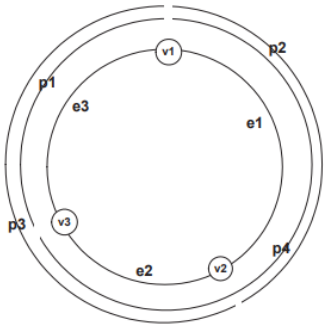


Fig.12: example for illustration. (See more in [1])

** :Commented [ma15]

### 5.2.1 General Topology:

**Lemma 2.1.** The competitive ratio of Online-MINADM is at least 7/4.

**Proof.** We will proof that the Online-MINADM is at least 7/4 by given a specific example. Let G be a cycle of three nodes V = {v1, v2, v3}, E = {e1, e2, e3} where

e1 = (v1, v2), e2 = (v2, v3), e3 = (v3, v1) and let P = {p1, p2, p3, p4} where p1 = (e3), p2 = (e1), p3 = (e2, e3), p4 = (e1, e2). Figure 6 shows the network and the paths. The optimal solution assigns w(p1) = w(p4) = λ1 and w(p2) = w(p3) = λ2, and uses 4 ADMs. Recall that Online-MINADM receives the paths of the input one at a time. It assigns w(p1) = λ1, then w(p2) = λ1 because λ1 is feasible for p2, then w(p3) = λ2 because λ1 is not feasible for p3 and finally w(p4) = λ3, because neither λ1 nor λ2 are feasible for p4. It uses 7 ADMs in total.

The proof demonstrates that the ONLINE-MINADM algorithm is not optimal when applied to a specific network configuration consisting of a cycle of three nodes and four paths. The optimal solution in this case uses 4 ADMs while the ONLINE-MINADM uses 7 ADMs. However, it should be noted that this is a specific case and the algorithm has been proven to be optimal in general, ring and path network configurations. The proof is provided for clarity and ease of understanding. Furthermore, S is the solution returned by the Online-MINADM and S* is the optimal solution.

**Lemma 2.2.** The competitive ratio of Online-MINADM is at most 7/4.

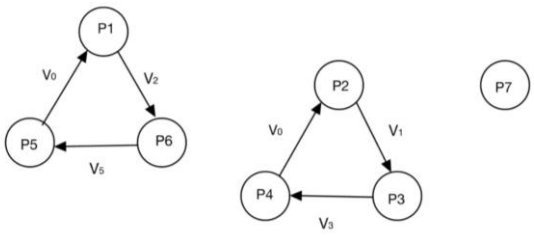


Fig.13: directed graph for the graph in Fig 9.

**Proof:**

Firstly, we direct each edge in the shareability graph of the optimal solution $g_{s*}$, such that each path becomes a directed path and each cycle become a directed cycle. The direction chosen for every path is arbitrary, like in **Fig 13.**

We denote to the directed shareability graph of the optimal solution by $\overrightarrow{g_{S_1^*}}$ .

For each p ∈ P we define $d_{in}(p)$ and $d_{out}(p)$ such that:

$d_{in}(p)$= the number of the edges that enters to it.

$d_{out}(p)$=the number of the edges that going out from it.

In this situation $d_{in}(p)\leq 1$ and $d_{out}(p) \leq 1$ for each node in $\overrightarrow{g_{S_1^*}}$ ,and that is because each node in $g_{s*}$ has $d(p) \leq 2$ so when we direct the graph, we separate d(p) into $d_{in}(p)$ and $d_{out}(p)$ such that $d_{in}(p) + d_{out}(p)=d(p)$.

The following definitions refer to $\overrightarrow{g_{S_1^*}}$ :

- $LAST^*$ is a set of nodes in $\overrightarrow{g_{S_1^*}}$ that have $d_{out}(p) =0$ so:

$$LAST^* = \{p \in P | d_{out}(p) = 0\}.$$

So, if p ∈$LAST^*$ that means that p is isolated node ($d_{in}(p) = d_{out}(p)=0$) or p is the last node (path) in a valid chain, thus $|LAST^*| = |P_{s*}|$.

- The functions $Next^*$ and $Prev^*$ are defined as expected: Next∗ maps a node p to the next node in $\overrightarrow{g_{S_1^*}}$ and $Prev^*$ maps a node p to the previous node in $\overrightarrow{g_{S_1^*}}$ whenever such a node exists, namely:

$$Next^*: P \setminus LAST^* \rightarrow P$$

If $d_{out}(p)$ is not zero, that means that it is equal to 1, therefore, $Next^*(p)$ is not empty and it is the unique node u such that there is an edge from p to u in $\overrightarrow{g_{S_1^*}}$.

Note: $Prev^* = Next^{*-1}$.

Now we partition the set of isolated nodes of $G_S$ ($D_0(S)$), into 4 sets A, B, C and D using the definitions above and the graphs $G_{S^*}$ and $\overrightarrow{g_{S_1^*}}$.

The classification procedure: **(CLASSIFY)**

**CLASSIFY** (p ∈ $D_0(S)$) {

If (p∈ LAST*) then {

p∈A; $f_A(p)=p$;

} else {

q= Next*(p);

if (q∈ $D_2(S)$ then {

p∈B; $f_B(p)=q$;

} else if (q∈ $D_1(S)$ then {

p∈C; $f_C(p)=\{p,q\}$ ;

} else {// q∈ $D_0(S)$

p∈D

}

}

}

From the code above we can understand that:

- A= {p∈P | p∈ LAST*}, this means that if p∈A, then p is an isolated node in the solution S (the solution of Online-MINADM algorithm) and p∈ LAST*.

- $f_A$: A → LAST*, therefore it is a one-to-one function because $f_A(p)=p$. In addition, **|A|≤ | LAST*|= $|P_{S^*}|$**, because if all the paths p∈ LAST* were isolated nodes in the solution S, then |A|= | LAST*|= $|P_{S^*}|$ and that is the upper bound for |A|.

- B= {p∈P | Next*(p)∈ $D_2(S)$}, this means that if p∈B, then p is an isolated node in the solution S and q= Next*(p)∈ $D_2(S)$. (q is not an isolated node in the solution S).

- $f_B$: B → $D_2(S)$, therefore it is a one-to-one function because $f_B(p)$= Next*(p)=q. Therefore **|B| ≤ $|D_2(S)| = d_2(S)$.**

- C= {p∈P | Next*(p)∈ $D_1(S)$}, this means that if p∈C, then p is an isolated node in the solution S and q= Next*(p)∈ $D_1(S)$. (q is not an isolated node in the solution S).

- $f_C$: C → $2^P$, and $f_C$= {p, q} where p ∈ $D_0(S)$ and q !∈ $D_0(S)$. in order to proof that $f_C$ is one to one, we assume that $f_C(p) \cap f_C(p')$ not empty set (assume that $f_C$ is not one to one). Let $f_C(p)$ = {p, q} and $f_C$ (p′) = {p′, q′}. Then p=p' or q=q'. In the latter case q = Next* (p) = Next* (p ′) = q ′, then p = p ′. Because from the definition of the directed graph, $d_{in}(q)= d_{in}(q')$ ≤1 there if p! = p' then $d_{in}(q)=d_{in}(q')$ =2, a contradiction. So, p=p',

and this means that if p! = p' then $f_C(p) \cap f_C(p')$ is an empty set, that mean, $f_C$ is one to one. As the sets $f_C$ (p) contain exactly 2 elements and because $f_c$ is one to one, **$|C| \leq \frac{N}{2}$**

- D= {p∈P | $Next^*(p) \in D_0(S)$}, this means that if $p \in D$, then p is an isolated node in the solution S and q= $Next^*(p) \in D_0(S)$. (q also is an isolated node in the solution S).

- here we show that D is an empty set. Assume, by contradiction that $p \in D$ for some $p \in D_0(S)$. Then there is $q \in D_0(S)$ such that $q = Next^*(p)$, therefore $(p, q) \in E_{S*} \subseteq E_\alpha$. Online-MINADM assigned unique colors to each of p and q. Assume without loss of generality that q comes later than p in the input sequence. p is assigned a unique color, therefore it is the only element in its chain. Then w(p) is feasible for q. Then the algorithm should assign w(q) = w(p), a contradiction. Then D = ∅.

According to the A, B, C, D like defined above and **definition 13** with **lemma 1.1**:

$d_0(S) = |D_0(S)| = |A| + |B| + |C| + |D| \leq |PS^*| + d_2(S) + \frac{N}{2}$. Then:

$$\varepsilon(\boldsymbol{S}) = \frac{\mathbf{d0}(\boldsymbol{S}) - \mathbf{d2}(\boldsymbol{S}) - \mathbf{2}\,|\mathbf{PS}*|}{\mathbf{N}} \leq \frac{1}{2}$$

$\varepsilon(\boldsymbol{S}) \leq \frac{1}{2}$ because is we replace the upper bound of $d_0(S) \leq |PS^*| + d_2(S) + \frac{N}{2}$:

$$\varepsilon(S) = \frac{|PS*| + d2(S) + \frac{N}{2} - d2(S) - 2\,|PS*|}{N} = \frac{\frac{N}{2} - |PS*|}{N} \leq \frac{\frac{N}{2}}{N} \leq \frac{1}{2}$$

Substituting this in **Lemma 1.1** and recalling that $cost(S^*) \geq N$ we get:

**$Cost(S) \leq Cost(S^*) + \frac{1}{2}N\,(1 + \frac{1}{2}) = Cost\,(S^*) + \frac{3}{4}N \leq \frac{7}{4}\,Cost(S^*)$.**

This upper bound stems from placing all the upper bounds of all the sets that we found to found the upper bound of $\varepsilon(\boldsymbol{S})$ and placing it in the equation in **lemma1.1.**

In conclusion, the upper bound of Online-MINADM is:

**$Cost(S) \leq \frac{7}{4}\,Cost(S^*)$.**

## 5.2.2 Ring Topology:

According to lemma 2.1 and lemma 2.2 and because of the ring topology is a special case of general topology, we can clearly find that the competitive ratio in this topology is at least $^{7}/_{4}$ (According to **lemma 2.1**), and the competitive ratio in this topology is at most $^{7}/_{4}$ (According to **lemma 2.2**). Therefore, the competitive ratio of Online-MINADM in ring topology is $^{7}/_{4}$.

## 5.2.3 Path topology:

**Lemma 2.3.** Online-MINADM is $^{3}/_{2}$ -competitive in path topology.

**Proof:**

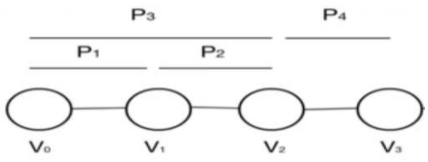


Fig.14. example of path topology.

Let V={$v_0$,$v_1$,$v_2$,$v_3$,$v_4$}, and P={$p_1$,$p_2$,$p_3$,$p_4$} like in **fig 14**. The paths ariavel is $p_1$,$p_2$,$p_3$ and $p_4$, for ech node vi we define to setts of paths $\sigma_i$ and $\tau_i$, such that $\sigma_i$ is set of paths that having $v_i$ as their right endpoint for example:

$\sigma_2$= {$p_2$, $p_3$}, because they end in $v_2$ from the right side.

$\tau_i$ is set of paths having $v_i$ as their left endpoint for example:

$\tau_2$={$p_4$} because it ends in $v_2$ from the left side.

The optimal number of ADMs in each node vi is max {$|\sigma_i|$ |, $|\tau_i|$} because we can assign one ADM for each pair, for example we can assign one ADM for pair $p_2$, $p_4$ in $v_2$. The number of the pairs in each node $v_i$ is min {$|\sigma_i|$, $|\tau_i|$}. And then we assign one ADM for the rest (max {$|\sigma_i$ |, $|\tau_i$ |} - min {$|\sigma_i$ |, $|\tau_i$ |}. So the optimal number of ADMs = min {$|\sigma_i$ | , $|\tau_i$ |} + max {$|\sigma_i$ | , $|\tau_i$ |} – min {$|\sigma_i$ | , $|\tau_i$ |} = max {$|\sigma_i$ | , $|\tau_i$ |}.Thus, the number of ADMs used by an optimal solution is $\Sigma_i$ max {$|\sigma_i$ | , $|\tau_i$ |}.

For each $v_i$ we define a maximum matching $MM_i$ that it is bipartite graph ($\sigma_i$, $\tau_i$, $\sigma_i$ × $\tau_i$).

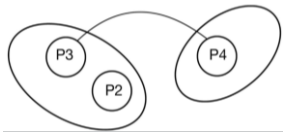

Fig.15. bipartite graph for v2.

The bipartite graph in **fig.15** is for $v_2$ such that the edge ($P_3$, $P_4$) means that the paths $p_3$, $p_4$ share one ADM, therefore the solution saves $|MM_i| = \min\{|\sigma_i|, |\tau_i|\}$ ADMs at node $v_i$, in other words $ES^* = \uplus_i MM_i$. Note that every matching of a complete bipartite graph can be augmented to a maximum matching.

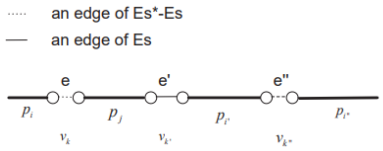


fig.16. Competitive ratio of Online-MINADM in the path topology. (See more in [1])

** :Commented [ma16]

Let S be the output of the algorithm and let S* be an optimal solution, such that the matching in each node is obtained by augmenting the matching done by S to a maximum matching, i.e., $ES \subseteq ES*$. We will now define a function **f** which maps from the set ($E_{S^*} \setminus E_S$) to $E_S$. Refer to **Figure 16** for further details.

We define $f(e) = e'$ if and only if there is $e = (p_i,p_j) \in E_{S^*} \setminus E_S$. $e \in E_{S^*} = \uplus_i MM_i$. Let $e \in MM_k$ for some node $v_k \in V$. Assume without loss of generality that i < j, i.e., path pi appears before pj in the input. As e not belong to $E_S$, none of $p_i$, $p_j$ are paired with any path at node $v_k$. Therefore, when $p_j$ appears in the input $w(p_i)$ is feasible for $p_j$, if it is not assigned color w(pi), this can be only because it is assigned color w(pj) = w(pi′), for some i′ < j. Let the common node of pj and pi ′ be vk′. Then e ′ = (pj,pi′) ∈ $E_S$. In another words, e′ is a substitutional edge for e in $E_S$.

Note: e' is uniquely defined because there is no other path besides $p_j$ and $p_{i'}$ that would get the same color and end at node $v_{k'}$, since $p_j$ is not paired at node $v_k$. As a result, k' must be different from k.

We claim that **f** is one-to-one, which means that each element in the range of the function must have a unique pre-image. To prove this, assume that there exists an element e'' which is not equal to e and f(e'') = e'. Since e'' is in the set ES* \ ES it must be in the set $MM_{k''}$ for some node $v_{k''}$. Since f was constructed in a certain way, k'' is the other endpoint of $p_{i'}$. Since e'' is equal to ($p_{i'}$, $p_{i''}$) it follows that j < i'. This contradicts our assumption, so we must conclude that f is one-to-one. so, $|E_{S^*}| - |E_S| = |E_{S^*} \setminus E_S| \leq |E_S|$, thus $|E_S| \geq 1/2\ |E_{S^*}|$.

We conclude as follows:

$$\text{Cost}(S) - \text{Cost}(S^*) = |E_{S^*}| - |E_S| \leq (|E_{S^*}|)/2 \leq \frac{N}{2} \leq \frac{\text{Cost}(S*)}{2}, \text{ therefore:}$$

$$\textbf{Cost(S)} \leq \frac{3}{2}\textbf{Cost(S*).}$$

## 5.3 Lower Bounds:

### 5.3.1 General Topology:

**Lemma 3.1:** There is no deterministic on-line algorithm with competitive ratio < $\frac{7}{4}$.

**proof**

we need to proof that for each deterministic on-line algorithm ALG with competitive ratio ρ, ρ ≥ $\frac{7}{4}$ .The proof uses a specific network diagram (Figure 17) and assigns colors to lightpaths using numbers 1, 2, etc. The first lightpath in the input is EFG and it is assumed that the algorithm assigns the color 1 to this lightpath. In other words w(EFG)=1.

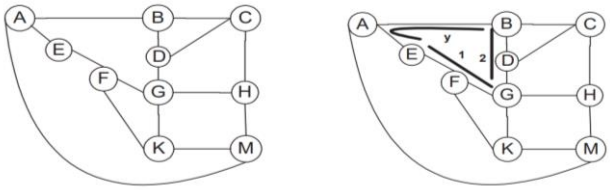


fig.17. A network used for the proof of the $\frac{7}{4}$ lower bound on general topology. (See more in [1])

** :Commented [ma17]

The second lightpath in the input is BDG. First assume that w(BDG) = 1. In this case if lightpath EABDG arrives, we have w(EABDG) = 2, then when lightpath GFEAB arrives we have w(GFEAB) = 3. ALG thus uses 7 ADMs, while it is easy to see the an optimal solution can use only 4 ADMs, thus ρ ≥ $\frac{7}{4}$, a contradiction. Hence, the algorithm ALG can't assign same color to BDG and EFG because if it did ρ ≥ $\frac{7}{4}$ thus w(BDG) = 2.

When the third lightpath in the input y=BAE arrives the situation is as depicted in the right hand side of Figure 17. It is clear that w(y) ≠ 3, since otherwise ρ ≥ $\frac{6}{3}$ > $\frac{7}{4}$ , a contradiction. Because if ALG assign for lightpath y new color (w(y)=3) the cost of ALG will be 2 * | P | = 2 *3 = 6, and the optimal solution is to give all the lightpaths the same color and close the circle, which means that the cost of the optimal solution is 3. Hence w(y)=1 or w(y)=2. Case a: w(y)=1.

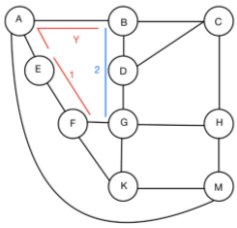

Fig.18. case a, w(y)=1.

Let z=EFKMHG be the next lightpath in the input sequence. Clearly w(z) ≠ 1. Hence w(z) = 2 or w(z) = 3

W(z)=2:

In this case, when lightpaths GFEAB, EABDG, BDGFE and EABCDG arrive, we get w(GFEAB) = 3, w(EABDG) = 4, w(BDGF E) = 5, w(EABCDG) = 6, and ρ = $\frac{14}{8} = \frac{7}{4}$ , a contradiction.

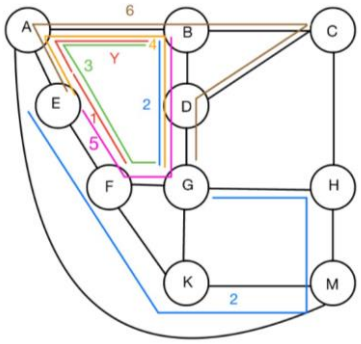


Fig.19. case a, w(y)=1 and W(z)=2.

w(z) = 3:

In this case, for u=EABDCHG we have w(u) = 4, and ρ ≥ $\frac{9}{5} > \frac{7}{4}$ , a contradiction.

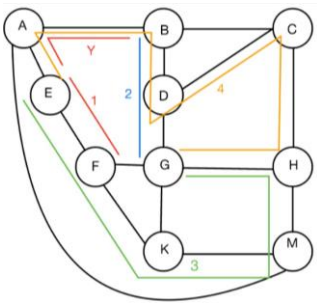


Fig.20. case a, w(y)=1 and W(z)=3.

case b:

w(y) = 2 Let z=BDCHG. Clearly w(z) ≠ 2. Hence w(z) = 1 or w(z) = 3.

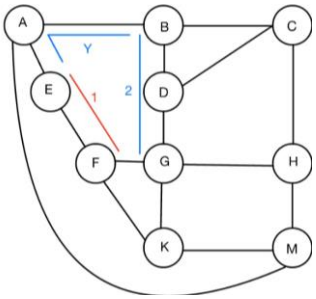


Fig.21. case b, w(y) = 2

w(z) = 1 When lightpaths EABDG, GFEAB, GKFEAB, and EFGDB arrive, we have w(EABDG) = 3, w(GF EAB) = 4, w(GKF EAB) = 5, w(EF GDB) = 6, and $\rho \geq \frac{14}{8} = \frac{7}{4}$ , a contradiction.

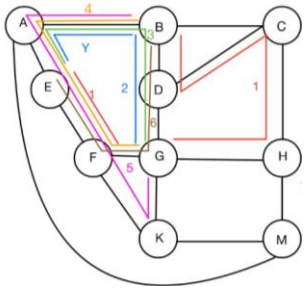


Fig.22. case b, w(y) = 2 and w(z) = 1

w(z) = 3 For u=GHMKFEAB we have w(u) = 4. Then $\rho \geq \frac{9}{5} > \frac{7}{4}$ , a contradiction.

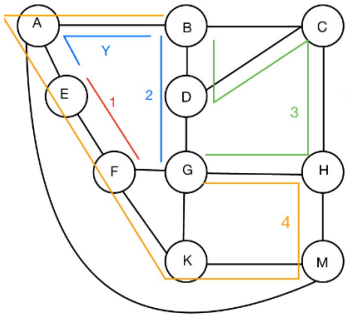


Fig.23. case b, w(y) = 2 and w(z) = 3

In Conclusion, as we see in all the above cases always the competitive ratio is ρ ≥ $\frac{7}{4}$ , Hence there is no deterministic on-line algorithm with competitive ratio < $\frac{7}{4}$.

### 5.3.2 Path Topology:

**Lemma 3.2:**

To prove that ONLINE-MINADM is optimal, we prove that there is no ($^3/_2$-ε)-competitive deterministic algorithm for path topology. Let's look at the following adversary:

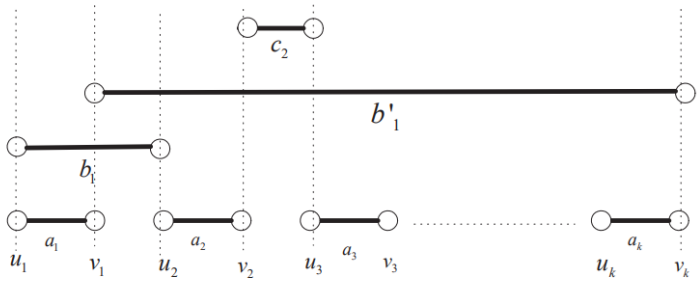


Fig.24. (See more in [1])

** :Commented [ma18]

Assume that we have k paths and $2k$ nodes like above. We will define ALG as the deterministic algorithm. in the first part our input is $a_1, a_2 \dots, a_k$ paths, every path i connects the nodes $u_i, v_i$ . in this stage the algorithm uses $2k$ ADMs, one for each node.


Fig.25.

Let's use this example. Here we have k=3. And paths $a_1, a_2, a_3$ with endpoints $(u_i, v_i)$ for $i = 1,2,3$ and ALG uses one ADM at each node, so we have 6 ADMs in this example.

In the second part our input depends on the decisions that ALG made in the first part. ALG making decision as follows, if the color of $a_i$ is the same as $a_{i+1}$ then it means

that the input contains 2 paths $b_i$ and $b_{i'}$. Where $b_i$ is the path that connects nodes $u_1, u_{i+1}$. And $b_{i'}$ connects the nodes $v_i, v_k$. In the example, we will have $b_1$ and $b_{1'}$.

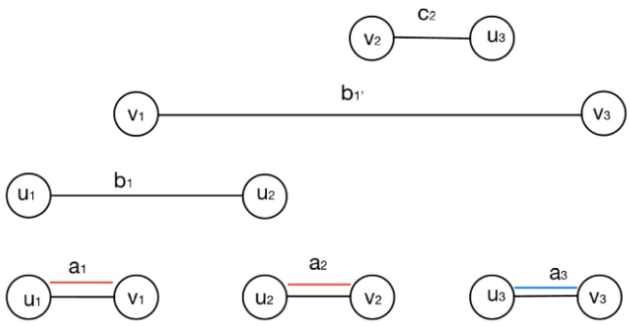


Fig.26.

For these paths, the color of $a_i$ and $a_{i+1}$ is not feasible for any of them because they are overlapping so it assigned a new color. As you can see, we can't use the same color as $a_1$ and $a_2$ so we assign it a new color as follows:

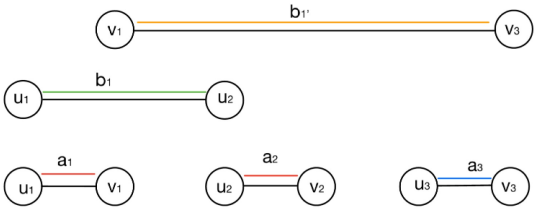


Fig.27.

The algorithm uses in $4$ ADMs that means, it is uses $4x$ ADMs where $x$ is the number of times that we have the same color in $a_i$ and $a_{i+1}$. As you can see ALG uses 2 ADMs for path $b_1$ and 2 ADMs for path $b_{1'}$. Otherwise, if $w\ (a_i)\ \neq w\ (a_{i+1})$ the input contains only one path $c_i$ that connects nodes $v_i, u_{i+1}$. See example:

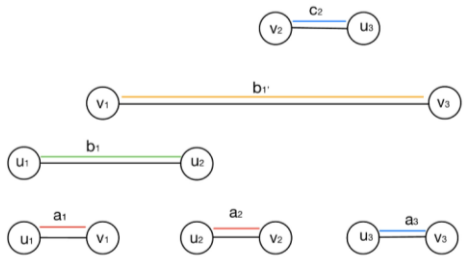


Fig.28.

We will define $x$ as the number of times that happened that the color of $a_i$ is the same as $a_{i+1}$ i. e., $w\ (a_i)\ =\ w\ (a_{i+1})$. So, it means that $k-1-x$ times we get that $w\ (a_i)\ \neq\ w\ (a_{i+1})$. It means that in the example when we have the same color for $a_1$ and $a_2$ , x will be 1 and we get that for the $a_2$ and $a_3$ we get that they have a different color. For the path $c_i$, if we color it as the same color as $a_i$ or $a_{i+1}$ this way the algorithm will use in only one ADM otherwise it will use in 2 ADMs. We can say that for paths $c_i$ the algorithm uses in at least $k-1-x$ ADMs. If we color $c_2$ in the same color as $a_2$ or $a_3$ we are going to use 1 ADM instead of 2. Let's say we choose the same color as $a_3$ so now we use 1 ADM for this path.

As we mentioned, in the first stage of the input we will get $2k$ ADMs and for paths $b_i$ and $b_i$, we will get $4x$ ADMs and for paths $c_i$ we will get $k-1-x$ ADMs. So, summing up we will get $2k+4x+k-1-x=3(k\ +\ x)-1$ ADMs. We will use our example and say that we use 6 ADMs for the first stage, and we use 4 for paths $b_1, b_1$, and 1 for path $c_2$ ending up with 11 ADMs.

But, on the other hand if for every consecutive paths $c_i, c_{i+1}, \ldots., c_{i+j}$ color such that $w(b_{i-1})=\ w(a_i)=\ w(c_i)=\ w(a_{i+1})=\ w(c_{i+1})=\ldots=\ w(c_{i+j})=\ w(a_{i+j+1})=\ w(b'_{i+j+1})$.

Let's look at the example and as you can see, we can use the same color for path $a_2, c_2, b_1, a_3$ and for paths $a_1, b_1$, another color and we end up using 8 ADMs. These solutions use $2k\ +\ 2x$ ADM's, one ADM at each $u_i, v_i$, $x$ additional ADMs at $u_1$, and x additional ADMs at $v_k$. Therefore, the competitive ratio of ALG is at least

$$\frac{3(k+x)-1}{2(k+x)}=\frac{3}{2}-\frac{1}{2(k+x)}\geq\frac{3}{2}-\frac{1}{2k}$$

When $3(k+x)-1$ is the worst case and $2(k+x)$ is the optimal case. For ε> 0, $k>\frac{1}{2\varepsilon}$ ,we get that the competitive ratio of ALG is higher then $\frac{3}{2}-\varepsilon$.

### 5.3.3 Ring Topology:

**Lemma 3.3** No deterministic on-line algorithm has a competitive ratio better than $\frac{7}{4}$, even for the ring topology.

As we did in the previous topology, we assume there is such algorithm that is denoted $ALG$ and see that such A does not exist. We look at the input of an adversary that intentionally chooses difficult input to maximize the ratio of the online algorithm and the optimal solution.

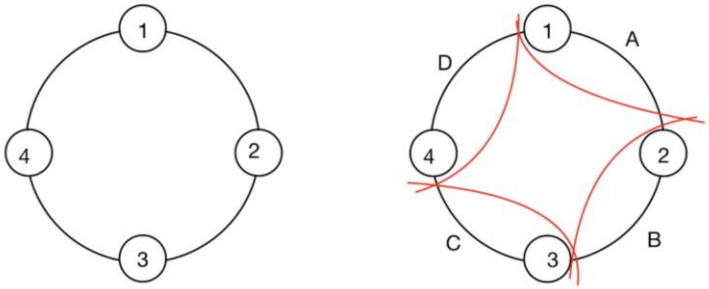


Fig.29. division into 4 segments

Assume we divide a ring network it into four segments A, B, C, D like described in **Fig 29**, where every segment consists of $^{N}/_{4}$ with their edges, N is the number of nodes in the network. Assume the adversary first requests lightpaths that are in segments that are not adjacent. A simple example would be the network at the left figure where there are only four nodes $V = \{1, 2, 3, 4\}$ that preform a ring, after the division we get the right figure, and the adversary requests lightpaths $A$ and $C$. Case 1 is where algorithm $ALG$ assigns the same color to $A$ and $C$, yet case 2 is where it assigns different colors.

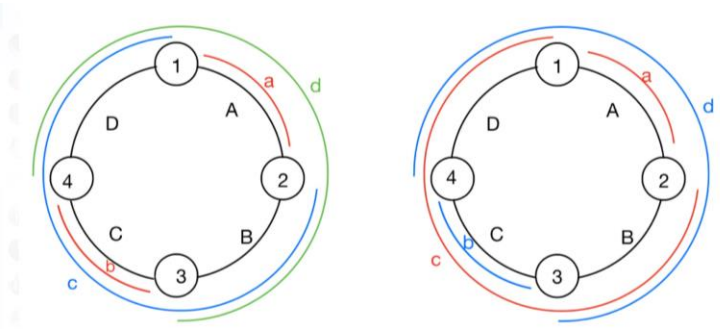


Fig.30. case 1, ALG uses 8 ADMs and the optimal is 4 ADMs

If algorithm $ALG$ assigns $A$ and $C$ the same color $\lambda_1 = red$ and next, lightpath $c$ arrives and it crosses segments $B, C,$ and $D$. After that lightpath $d$ arrives crossing segments $D, A$, and $B$. $c$ and $d$ are assigned different colors each $\lambda_2 = blue$, $\lambda_3 = green$ respectively since $c$ is conflicts path $b$ and path $d$ conflicts paths $a$ and $c$. In this case, we use 8 ADMs, like described in the left side of **Fig 30**, two for each lightpath in the input. The optimal offline solution uses 4 ADMs like described in the right side of **Fig 30** thus, the competitive ratio is $^8/_4 > {}^7/_4$. This means that algorithm $ALG$ assigns the same color to lightpaths in segment $A$ and $C$ will not perform better than ONLINE-MINADM.

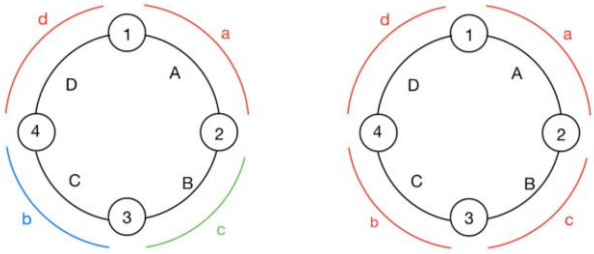


Fig.31. case 2, ALG uses 7 ADMs and the optimal is 4 ADMs

In case 2, algorithm $A$ assigns different colors $\lambda_1 = red$, $\lambda_2 = blue$ to $A$ and $C$, assume the next request is $B$ and $ALG$ assigns to it a new color $\lambda_3 = green$, if the request after that is $D$, $A$ assigns to it $\lambda_1 = red$ since $red$ is feasible for $D$. This way the network uses 7 ADMs, like described in the left side of **Fig 31**, while the optimal offline solution uses 4 ADMs, like described in the right side of **Fig 31**. The competitive ratio in this case is $^7/_4$ and not less than the competitive ratio of algorithm MIN-ADM. For conclusion, in both cases the competitive ratio of algorithm $ALG$ is not less than $^7/_4$.

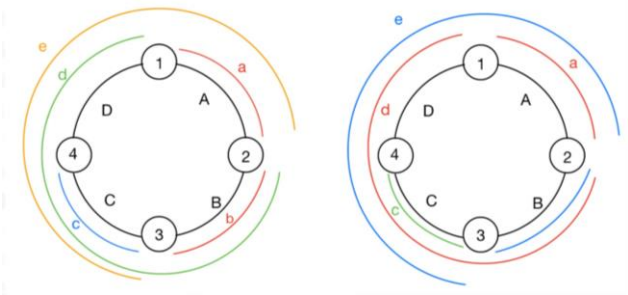


Fig.32. case 3, ALG uses 9 ADMs and the optimal is 6 ADMs

In case 3, algorithm $A$ assigns the same color $\lambda_1 = red$ to $A$ and $B$, and a different color $\lambda_2 = blue$ to $C$. If the adversary requests lightpaths $c$ and $d$ that cross $B$, $C$, $D$ and $C$, $D$, $A$ respectively, then both will get new colors $\lambda_3 = green$, $\lambda_4 = yellow$ since they are conflicting with the other paths. In this case, $ALG$ uses 7 ADMs for $a, b, d$ and $e$, and 2 more for $C$ ,like described in the left side of **Fig 32**. The optimal offline solution uses 4 ADMs for $a, b, d$ and $e$, and 2 more for $C$, like described in the right side of **Fig 32**. If the adversary now requests again the process for k times, $ALG$ will use $7k + 2$ ADMs and the offline will use $4k + 2$ ADMs. Repeating the process makes sure that future lightpaths in the input use new colors. In this case the competitive ratio is $^{7\mathrm{k}+2}/_{4\mathrm{k}+2}$ so it is at least $^{7}/_{4} - \epsilon$ for any $\epsilon > 0$. For conclusion, there is no such deterministic online algorithm A that has a competitive ratio better than the competitive ratio of algorithm ONLINE-MINADM.

## 6. Evaluation plan:

In the second part of the project, we will simulate the proposed online algorithm in various network topologies, specifically in ring and path configurations. The main objectives of this simulation are twofold: First, to visually demonstrate the execution of the algorithm, and second, to evaluate its expected performance. Typically, the analysis of an online algorithm focuses on its worst-case competitive ratio, which compares its performance to the optimal algorithm. However, in practical scenarios, one may also be interested in evaluating the expected cost compared to the best possible outcome.

## 7.Summary:

In this project, we discussed the topic of optimal networks and online algorithms and presented an optimal online algorithm, first introduced in the book, with detailed examples to enhance understanding. This algorithm has a competitive ratio of 7/4 in general and ring topologies, and 3/2 in path topology. We examined each topology and established upper and lower bounds of the algorithm. Next, we plan to develop a software to visualize the algorithm and analyze its average case, as an addition to the existing research.

## References:

[1] Shalom, M., Wong, P. W., & Zaks, S. (2007, September). Optimal on-line colorings for minimizing the number of ADMs in optical networks. In *International Symposium on Distributed Computing* (pp. 435-449). Springer, Berlin, Heidelberg.

[2] Borodin, A., & El-Yaniv, R. (2005). *Online computation and competitive analysis*. cambridge university press.

[3] Gollapudi, S., & Panigrahi, D. (2019, May). Online algorithms for rent-or-buy with expert advice. In *International Conference on Machine Learning* (pp. 2319-2327). PMLR.

[4] Technical Papers and Presentations: Dense Wavelength-Division Multiplexing (DWDM) (topicstash.blogspot.com)

[5] Part 8: WDM SYSTEM. - ppt video online download (slideplayer.com)