Maintaining Perfect Matchings at Low Cost

Jannik Matuschke\(^1\), Ulrike Schmidt-Kraepelin\(^2\), and José Verschae\(^3\)

\(^1\) Technische Universität München
\(^2\) Technische Universität Berlin
\(^3\) Universidad de O’Higgins

Abstract. The min-cost matching problem suffers from being very sensitive to small changes of the input. Even in a simple setting, e.g., when the costs come from the metric on the line, adding two nodes to the input might change the optimal solution completely. On the other hand, one expects that small changes in the input should incur only small changes on the constructed solutions, measured as the number of modified edges.

We introduce a two-stage model where we study the trade-off between quality and robustness of solutions. In the first stage we are given a set of nodes in a metric space and we must compute a perfect matching. In the second stage \(2k\) new nodes appear and we must adapt the solution to a perfect matching for the new instance.

We say that an algorithm is \((\alpha, \beta)\)-robust if the solutions constructed in both stages are \(\alpha\)-approximate with respect to min-cost perfect matchings, and if the number of edges deleted from the first stage matching is at most \(\beta k\). Hence, \(\alpha\) measures the quality of the algorithm and \(\beta\) its robustness. In this setting we aim to balance both measures by deriving algorithms for constant \(\alpha\) and \(\beta\).

We show that there exists an algorithm that is \((3, 1)\)-robust for any metric if one knows the number \(2k\) of arriving nodes in advance. For the case that \(k\) is unknown the situation is significantly more involved. We study this setting under the metric on the line and devise a \((19, 2)\)-robust algorithm that constructs a solution with a recursive structure that carefully balances cost and redundancy.

1 Introduction

Weighted matching is one of the founding problems in combinatorial optimization, playing an important role in the settling of the area. The work by Edmonds [8] on this problem greatly influenced the role of polyhedral theory on algorithm design [23]. On the other hand, the problem found applications in several domains [20, 11, 22, 16, 12]. In particular routing problems are an important area of application, and its procedures often appeared as subroutines of other important algorithms, the most notable being Christofides’ algorithm for the traveling salesperson problem [5].

An important aspect of devising solution methods for optimization problems is studying the sensitivity of the solution towards small changes in the input. This sensitivity analysis has a long history and plays an important role in practice [10]. Min-cost matching is a problem that has particularly sensitive optimal solutions.
Assume for example that nodes lie on the real line at points $\ell$ and $\ell + 1 - \varepsilon$ for some $0 < \varepsilon < 1$ and all $\ell \in \{1, \ldots, n\}$, see Fig. 1. The min-cost matching, for costs equal the distance on the line, is simply the edges $\{\ell, \ell + 1 - \varepsilon\}$. However, even under a minor modification of the input, e.g., if two new nodes appear at points $1 - \varepsilon$ and $n + 1$, the optimal solution changes all of its edges, and furthermore the cost decreases by a $\Theta(1/\varepsilon)$ factor. Rearranging many edges in an existing solution is often undesirable and may incur large costs, for example in an application context where the matching edges imply physical connections or binding commitments between nodes. A natural question in this context is whether we can avoid such a large number of rearrangements by constructing a robust solution that is only slightly more expensive. In other words, we are interested in studying the trade-off between robustness and the cost of solutions.

We consider a two-stage robust model with recourse. Assume we are given an underlying metric space $(X, c)$. The input for the first stage is a complete graph $G_1$ whose node set $V(G_1)$ is a finite, even subset of $X$. The cost of an edge $\{v, w\}$ is given by the corresponding cost $c(v, w)$ in the metric space. In a second stage we get an extended complete graph $G_2$ containing all nodes in $V(G_1)$ plus $2k$ additional nodes. As before, costs of edges in $G_2$ are given by the underlying metric. In the first stage we must create a perfect matching $M_1$ for $G_1$. In the second stage, after $G_2$ is revealed, we must adapt our solution by constructing a new perfect matching $M_2$ for $G_2$, called the second stage reply. We say that a solution $M_1$ is two-stage $(\alpha, \beta)$-robust if for any instantiation of the second stage there exists a solution $M_2$ such that two conditions hold. First, the total cost of edges in $M_i$ must satisfy $c(M_i) \leq \alpha \cdot c(O_i)$ for $i \in \{1, 2\}$, where $O_i$ denotes a min-cost perfect matching in $G_i$. Second, it must hold that $|M_1 \setminus M_2| \leq \beta k$.

An algorithm is two-stage $(\alpha, \beta)$-robust if, given $G_1$ and $c$, it returns a two-stage $(\alpha, \beta)$-robust matching and, given the set of new arrivals, a corresponding second stage reply. We refer to $\alpha$ as the competitive factor and $\beta$ as the recourse factor of the algorithm. Our main goal is to balance cost and recourse, and thus we aim to obtain algorithms where $\alpha$ and $\beta$ are constants.

Our model is closely related to an online model with recourse. Consider a graph whose nodes are revealed online two by two. Our objective is to maintain a perfect matching at all times. As above, irrevocable decisions do not allow for constant competitive factors. This suggests a model where in each iteration we are allowed to modify a constant number of edges. An $\alpha$-competitive algorithm that deletes at most $\beta$ edges per iteration can be easily transformed into a two-stage $(\alpha, \beta)$-robust algorithm. Thus, we can think that our two-stage model is a first step for understanding this more involved online model.
Our Results and Techniques. We distinguish two variants of the model. In the \(k\)-known case we assume that in Stage 1 we already know the number of new nodes \(2k\) that will arrive in Stage 2. For this case we present a simple two-stage \((3,1)\)-robust algorithm.

Theorem 1. Let \((X, c)\) be a metric space, \(V_1 \subseteq X\) with \(|V_1|\) even, and \(G_1\) be the complete graph on \(V_1\). For \(k \in \mathbb{N}\) known in advance, there is a perfect matching \(M_1\) in \(G_1\) that is two-stage \((3,1)\)-robust for \(2k\) arrivals. Such a matching and corresponding second stage reply can be computed in time poly\((|V_1|, k)\).

The example in Fig. 1 illustrates a worst case scenario for the strategy of choosing \(O_1\) as the first stage matching for \(k = 1\). The reason for this is that the nodes arriving in Stage 2 induce a path in \(O_1 \Delta O_2\) that incurs a significant drop in the optimum value. Our algorithm is designed towards preparing for such bad scenarios. To this end, we define the notion of gain for a path \(P\) with respect to a matching \(X\) as follows:

\[
\text{gain}_X(P) := c(P \cap X) - c(P \setminus X).
\]

In Stage 1, our algorithm chooses \(k\) disjoint \(O_1\)-alternating paths of maximum total gain with respect to \(O_1\). For each such path \(P\) we modify \(O_1\) by removing \(P \cap O_1\) and adding \((P \setminus O_1) \cup \{e(P)\}\), where \(e(P)\) is the edge that connects the endpoints of \(P\). Our choice of paths of maximum gain implies that \(P \cap O_1\) is larger than \(P \setminus O_1\). Therefore we can bound the cost of the solution in the first stage against that of \(O_1\) and also infer that most of its costs is concentrated on the edges \(e(P)\). For the second stage we construct a solution for the new instance by removing the \(k\) edges of the form \(e(P)\) and adding new edges on top of the remaining solution. The algorithm is described in detail in Section 2.

For the case where \(k\) is unknown the situation is considerably more involved as a first stage solution must work for any number of arriving nodes simultaneously. In this setting we restrict our study to the real line and give an algorithm that is two-stage \((19, 2)\)-robust.

Theorem 2. Let \(X = \mathbb{R}\) and \(c = \| \cdot \|\), \(V_1 \subseteq X\) with \(|V_1|\) even, and let \(G_1\) be the complete graph on \(V_1\). Then there is a perfect matching \(M_1\) in \(G_1\) that is two-stage \((19, 2)\)-robust. Such a matching, as well as the second stage reply, can be computed in time poly\((|V_1|, k)\).

The first stage solution \(M\) is constructed iteratively, starting from the optimal solution. We will choose a path \(P\) greedily such that it maximizes \(\text{gain}_M(P)\) among all alternating paths that are heavy, i.e., the cost of \(P \cap M\) is a factor 2 more expensive than the cost of \(P \setminus M\). Then \(M\) is modified by augmenting along \(P\) and adding edge \(e(P)\), which we fix to be in the final solution. We iterate until \(M\) only consists of fixed edges. As we are on the line, each path \(P\) corresponds to an interval and we can show that the constructed solution form a laminar family. Furthermore, our choice of heavy paths implies that their lengths satisfy an exponential decay property. This allows us to bound cost of the first
stage solution. For the second stage, we observe that the symmetric difference $O_1 \Delta O_2$ induces a set of intervals on the line. For each such an interval, we remove on average at most two edges from the first stage matching and repair the solution with an optimal matching for the exposed vertices. A careful choice of the removed edges, together with the greedy construction of the first stage solution, give us constant factor guarantees for the total cost of $O_2$-edges inside these intervals. We can use this to argue that the cost of the resulting second stage solution is within a constant factor of the optimum. See Sections 3 and 4 for a detailed description of this case.

Related Work. Intense research has been done on several variants of the online bipartite matching problem [17,16,18,4,21]. In this setting we are given a known set of servers while a set of clients arrive online. In the online bipartite metric matching problem servers and clients correspond to points from a metric space. Upon arrival, each client must be matched to a server irrevocably, at cost equal to their distance. For general metric spaces, there is a tight bound of $(2n - 1)$ on the competitiveness factor of deterministic online algorithms, where $n$ is the number of servers [18,16]. Recently, Raghvendra presented a deterministic algorithm [24] with the same competitiveness factor, that in addition is $O(\log(n))$-competitive in the random arrival model. Also, its analysis can be parameterized for any metric space depending on the length of a TSP tour and its diameter [21]. For the special case of the metric on the line, Raghvendra [25] recently refined the analysis of the competitive ratio to $O(\log(n))$. This gives a deterministic algorithm that matches the previously best known bound by Gupta and Lewi [15], which was attained by a randomized algorithm. As the lower bound of 9.001 [9] could not be improved for 20 years, the question whether there exists a constant competitive algorithm for the line remains open.

The online matching with recourse problem considers an unweighted bipartite graph. Upon arrival, a client has to be matched to a server and can be reallocated later. The task is to minimize the number of reallocations under the condition that a maximum matching is always maintained. The problem was introduced by Grove, Kao and Krishnan [11]. Chaudhuri et al. [4] showed that for the random arrival model a simple greedy algorithm uses $O(n \log(n))$ reallocations with high probability and proved that this analysis is tight. Recently, Bernstein, Holm and Rotenberg [3] showed that the greedy algorithm needs $O(n \log^2 n)$ allocations in the adversarial model, leaving a small gap to the lower bound of $O(n \log n)$. Gupta, Kumar and Stein [14] consider a related problem where servers can be matched to more than one client, aiming to minimize the maximum number of clients that are assigned to a server. They achieve a constant competitive factor server while doing in total $O(n)$ reassignments.

Online min-cost problems with reassignments have been studied in other contexts. For example in the online Steiner tree problem with recourse a set of points on a metric space arrive online. We must maintain Steiner trees of low cost by performing at most a constant (amortized) number of edge changes per iteration. While the pure online setting with no reassignment only allows for
\( \Omega(\log(n)) \) competitive factors, just one edge deletion per iteration is enough to obtain a constant competitive algorithm [12]; see also [13,20].

The concept of recoverable robustness is also related to our setting [19]. In this context the perfect matching problem on unweighted graphs was considered by Dourado et. al. [7]. They seek to find perfect matchings which, after the failure of some edges, can be recovered to a perfect matching by making only a small number of modifications. They establish computational hardness results for the question whether a given graph admits a robust recoverable perfect matching.

## 2 Known Number of Arrivals

In this section, we consider the setting where \( k \) is already known in Stage 1. Let \( G_1 \) be the graph given in Stage 1 (with edge costs \( c \) induced by an arbitrary metric) and let \( O_1 \) be a min-cost perfect matching in \( G_1 \). Without loss of generality assume that \( |O_1| > k \), as otherwise, we can remove all edges of \( M_1 \) in Stage 2.

**Algorithm 1.1** works as follows: (i) Let \( P_1, \ldots, P_k \) be disjoint, \( O_1 \)-alternating paths maximizing \( \sum_{i=1}^{k} \text{gain}_{O_1}(P_i) \). (ii) Set \( \overline{M} := O_1 \Delta P_1 \Delta \cdots \Delta P_k \). (iii) Return \( M_1 := \overline{M} \cup \{ e(P_i) : i \in [k] \} \).

It is easy to see that each path \( P_i \) starts and ends with an edge from \( O_1 \) and \( \text{gain}_{O_1}(P_i) \geq 0 \). As a consequence, \( M_1 \) is a perfect matching and

\[
\text{cost}(\overline{M}) = \text{cost}(O_1) - \sum_{i=1}^{k} \text{gain}_{O_1}(P_i) \leq \text{cost}(O_1).
\]

Using \( \text{cost}(e(P_i)) \leq \text{cost}(P_i) \) and \( \bigcup_{i=1}^{k} P_i = O_1 \Delta \overline{M} \subseteq O_1 \cup \overline{M} \) we obtain

\[
\text{cost}(M_1) \leq \text{cost}(\overline{M}) + \sum_{i=1}^{k} \text{cost}(P_i) \leq \text{cost}(\overline{M}) + \text{cost}(\overline{M}) + \text{cost}(O_1) \leq 3 \times \text{cost}(O_1).
\]

Now consider the arrival of 2\( k \) new vertices, resulting in the graph \( G_2 \) with min-cost matching \( O_2 \). Note that \( O_2 \Delta \overline{M} \) is a \( U \)-join, where \( U \) is the set of endpoints of the paths \( P_1, \ldots, P_k \) and the 2\( k \) newly arrived vertices.

**Algorithm 1.2** works as follows: (i) Let \( P'_1, \ldots, P'_{2k} \) be the 2\( k \) maximal paths from \( O_2 \Delta \overline{M} \). (ii) Return \( M_2 := \overline{M} \cup \{ e(P'_i) : i \in [2k] \} \).

Note that \( O_1 \Delta O_2 \) consists of \( k \) alternating paths \( R_1, \ldots, R_k \), from which we remove the starting and ending \( O_2 \)-edge. Then these paths would have been a feasible choice for \( P'_1, \ldots, P'_k \), implying that the total gain of the \( R_i \)’s is at most that of the \( P'_i \)’s. We conclude that

\[
\text{cost}(\overline{M}) = \text{cost}(O_1) - \sum_{i=1}^{k} \text{gain}_{O_1}(P_i) \leq \text{cost}(O_1) - \sum_{i=1}^{k} \text{gain}_{O_1}(R_i) \leq \text{cost}(O_2).
\]

Applying \( \bigcup_{i=1}^{2k} P'_i \subseteq O_2 \Delta \overline{M} \subseteq O_2 \cup \overline{M} \), we obtain

\[
\text{cost}(M_2) \leq \text{cost}(\overline{M}) + \sum_{i=1}^{2k} \text{cost}(P'_i) \leq \text{cost}(\overline{M}) + \text{cost}(\overline{M}) + \text{cost}(O_2) \leq 3 \times \text{cost}(O_2).
\]

As \( |M_1 \setminus M_2| = |M_2 \setminus \overline{M}| = k \), we conclude that \( M_1 \) is indeed two-stage (3,1)-robust. We remark that the matchings described in the section can be computed efficiently by solving a minimum weight \( T \)-join problem in an extension of \( G_1 \). This concludes our proof of Theorem 1.
Fig. 2. Given $\alpha, \beta \in O(1)$, $G_1$ is constructed such that $O_1$ contains $\beta + 1$ edges of size $c_1$ and $\Omega(\beta^3)$ of size $c_3$ that are equally distributed between $c_1$-edges. The distance between any two edges in $O_1$ is $c_2$. Values $c_1, c_2, c_3$ are chosen depending on $\alpha$, guaranteeing $c_1 \gg \beta^3 c_2 \gg \beta^6 c_3$. Consider Algorithm 1.1 with $k = \beta + 1$. It chooses $O_1$ which is vulnerable to the arrival of 2 nodes at the extremes of the line, decreasing cost to $\Theta(\beta^3 c_2)$ and allowing $\beta$ deletions only. Consider Algorithm 1.1 with $k \leq \beta$. Its matching contains more than $\beta^2 + \beta$ edges of size $c_2$. Hence, it is vulnerable to the arrival of $2(\beta + 1)$ nodes that decrease the cost of an optimal solution to $\Theta(\beta^3 c_3)$ and allowing $\beta^2 + \beta$ deletions only.

3 Unknown Number of Arrivals – Stage 1

In this section, we consider the case that the underlying metric corresponds to the real line. This implies that there is a Hamiltonian path $L$ in $G_1$ such that $c(v, w) = c(L[v, w])$ for all $v, w \in V(G_1)$, where $L[v, w]$ is the subpath of $L$ between nodes $v$ and $w$. We will refer to $L$ as the line and call the subpaths of $L$ intervals. The restriction to the metric on the line results in a uniquely defined min-cost perfect matching $O_1$ with a special structure.

Lemma 3. $O_1$ is the unique perfect matching contained in $L$.

When the number of arrivals is not known in the first stage, the approach for constructing the first stage matching introduced in Section 2 does not suffice anymore. Fig. 2 illustrates a class of instances for which Algorithm 1.1 cannot achieve $(O(1), O(1))$-robustness, no matter how we choose $k$. For a matching $M$, define $g(M) := \max_{e \in L} |\{(v, w) \in M : e \in L[v, w]\}|$. The example in Fig. 2 can be generalized to show that we cannot restrict ourselves to constructing matchings $M_1$ with the property that $g(M_1)$ is bounded by a constant.

In view of the above example, we adopt the approach from Section 2 as follows. Instead of creating a fixed number of paths, our algorithm now iteratively and greedily selects a path $P$ of maximum gain with respect to a dynamically changing matching $X$ (initially $X = O_1$). In order to bound the total cost incurred by adding edges of the form $e(P)$, we only consider paths $P$ for which $X \cap P$ contributes a significant part to the total cost of $P$.

Definition 4. Let $X, P \subseteq E(G_1)$.

1. We say that $P$ is $X$-heavy if $c(P \cap X) \geq 2 \cdot c(P \setminus X)$.
2. We say that $P$ is $X$-light if $c(P \cap X) \leq \frac{1}{2} \cdot c(P \setminus X)$.

Algorithm 2.1 works as follows: (i) Initialization: Set $M_1 := \emptyset$ and $X := O_1$.  
(ii) While $X \neq \emptyset$: Let $P$ be an $X$-heavy $X$-alternating path maximizing $\text{gain}_X(P)$ and update $M_1 \leftarrow M_1 \cup \{e(P)\}$ and $X \leftarrow X \Delta P$.  
(iii) Return $M_1$. 

Hence, we obtain that there exist an even number of iterations $i$ such that $X \cap P^{(i)}$ was not modified between iteration $i$ and iteration $j$. Then, extending $P^{(i)}$ with the rightmost edge yields an $X^{(i)}$-heavy path with higher gain than $P^{(i)}$, a contradiction.

Note that in each iteration, the path $P$ starts and ends with an edge from $X$ as it is gain-maximizing (if $P$ ended with an edge that is not in $X$, we could simply remove that edge and obtain a path of higher gain). Therefore it is easy to see that $X \cup M_1$ is always a perfect matching, and in each iteration the cardinality of $X$ decreases by 1.

Now number the iterations of the while loop in Algorithm 2.1 from 1 to $n$. Let $X^{(i)}$ be the state of $X$ at the beginning of iteration $i$. Let $P^{(i)}$ be the path chosen in iteration $i$ and let $e^{(i)} = e(P^{(i)})$ be the corresponding edge added to $M$. The central result in this section is that the paths $P^{(i)}$ form a laminar family of intervals on the line. See Fig. 3 for an illustration of the proof idea and Appendix A.2 for the complete proof.

**Lemma 5.** 1. $X^{(i)}, P^{(i)} \subseteq L$ for all $i \in [n]$.
2. For all $i, j \in [n]$ with $i < j$, either $P^{(i)} \cap P^{(j)} = \emptyset$ or $P^{(j)} \subset P^{(i)}$.

**Tree structure.** Lemma 5 induces a tree structure on the paths selected by Algorithm 2.1. We define the directed tree $T = (W, A)$ as follows. We let $W := \{0, \ldots, n\}$ and define $P^{(0)} := L$. For $i, j \in W$ we add the arc $(i, j)$ to $A$ if $P^{(j)} \subset P^{(i)}$ and there is no $i' \in W$ with $P^{(j)} \subset P^{(i')} \subset P^{(i)}$. It is easy to see that $T$ is an out-tree with root 0. We let $T[i]$ be the unique 0-$i$-path in $T$. We define the set of children of $i \in W$ by $\text{ch}(i) := \{j \in W : (i, j) \in A\}$. Furthermore, let $W_H := \{i \in W : |T[i]| \text{ is odd}\}$ and $W_L := \{i \in W : |T[i]| \text{ is even}\}$ be the set of heavy and light nodes in the tree, respectively. These names are justified by the following lemma. See Fig. 4(a)-b) for an illustration.

**Lemma 6.** If $i \in W_H$, then $P^{(i)} \cap X^{(i)} = P^{(i)} \cap O_1$ and, in particular, $P^{(i)}$ is $O_1$-heavy. If $i \in W_L \setminus \{0\}$, then $P^{(i)} \cap X^{(i)} = P^{(i)} \setminus O_1$ and, in particular, $P^{(i)}$ is $O_1$-light.

**Proof.** Let $i \in W \setminus \{0\}$. From Lemma 5 we know that for every iteration $i' \in W \setminus \{0\}$ with $i' < i$ it holds that either $P^{(i') \cap P^{(i')}} = \emptyset$ or $P^{(i')} \subset P^{(i)}$. In the first case it holds that $P^{(i)} \cap X^{(i'+1)} = P^{(i)} \cap X^{(i')}$, in the latter case it holds that $P^{(i)} \cap X^{(i'+1)} = P^{(i)} \cap (P^{(i)} \Delta X^{(i')})$. Moreover, it is easy to see that $i' < i$ and $P^{(i')} \subset P^{(i)}$ holds if and only if $i' \in V(T[i]) \setminus \{0, i\}$. If $i \in W_H$, this implies that there exist an even number of iterations $i' < i$ for which $P^{(i')} \subset P^{(i)}$ holds. Hence, we obtain

$$P^{(i)} \cap X^{(i)} = P^{(i)} \cap (P^{(i)} \Delta \cdots \Delta P^{(i)} \Delta O_1) = P^{(i)} \cap O_1.$$
Note that a) Illustration of the matching created by an example execution of Algorithm 2.1. Edges added to $M_1$ in an iteration from $W_H$ are depicted by blue lines and edges created in an iteration from $W_L$ are illustrated by red dotted lines. b) Illustration of the corresponding tree. For every tree-node $i \in W$, grey edges indicate $X^{(i)}$ and an arc illustrates the edge connecting the end nodes of $P^{(i)}$. c) Illustration of example assignment of requests to iterations (defined in Section 4). Requests $R, R', R'' \in \mathcal{R}$ are assigned such that $R, R'' \in \mathcal{R}(0)$ and $R' \in \mathcal{R}(2)$. $\bar{R} \in \mathcal{R}(0)$ is a gap between two requests associated with tree-node 0.

If $i \in W_L$, this implies that there exist an odd number of iterations $i' < i$ for which $P^{(i)} \subset P^{(i')}$ holds. Hence, we can deduce that

$$P^{(i)} \cap X^{(i)} = P^{(i)} \cap \sum_{\text{oddly often}} (P^{(i)} \Delta \cdots \Delta P^{(i)} \Delta O_1) = P^{(i)} \setminus O_1. \quad \Box$$

The fact that nested paths are alternatingly $O_1$-heavy and $O_1$-light implies an exponential decay property. As a consequence we can bound the cost of $M_1$.

**Lemma 7.** Let $i \in W \setminus \{0\}$. Then $\sum_{j \in \text{ch}(i)} c(P^{(j)}) \leq \frac{1}{2} \cdot c(P^{(i)})$.

**Proof.** Let $i \in W \setminus \{0\}$. Then

$$\sum_{j \in \text{ch}(i)} f(P^{(j)}) \leq \frac{1}{2} \sum_{j \in \text{ch}(i)} c(P^{(j)}) \cap X^{(j)} \leq \frac{1}{2} c(P^{(i)} \setminus X^{(i)}) \leq \frac{1}{2} c(P^{(i)}),$$

where the first inequality follows from the fact that $P^{(j)}$ is $X^{(j)}$-heavy; the second inequality follows from the fact that $P^{(j)} \cap X^{(j)} \subseteq P^{(i)} \setminus X^{(i)}$ for $j \in \text{ch}(i)$ and the fact that the intervals $P^{(j)}$ for all children are disjoint; the last inequality follows from the fact that $P^{(i)}$ is $X^{(i)}$-heavy. \quad \Box

**Lemma 8.** $c(M_1) \leq 3c(O_1)$.

**Proof.** Note that $c(M_1) = \sum_{i = 1}^{n} c(e^{(i)}) = \sum_{i \in W \setminus \{0\}} c(P^{(i)})$. For $\ell \in \mathbb{N}$, let $W_{\ell} := \{i \in W : |T[i]| = \ell\}$.

Observe that Lemma 7 implies that $\sum_{i \in W_{\ell}} c(P^{(i)}) \leq \left(\frac{1}{2}\right)^{\ell-1} \sum_{i \in W_{\ell}} c(P^{(i)})$ for all $\ell \in \mathbb{N}$. Furthermore $\sum_{i \in W_{1}} c(P^{(i)}) \leq \frac{1}{2} c(O_1)$, because $W_1 \subseteq W_H$. Hence

$$c(M) = \sum_{\ell = 1}^{\infty} \sum_{i \in W_{\ell}} c(P^{(i)}) \leq \sum_{\ell = 1}^{\infty} \left(\frac{1}{2}\right)^{\ell-1} \sum_{i \in W_{\ell}} c(P^{(i)}) = 2 \cdot \frac{3}{2} c(O_1). \quad \Box$$
4 Unknown Number of Arrivals – Stage 2

We now discuss how to react to the arrival of $2k$ additional vertices. We let $O_2$ be the min-cost perfect matching in the resulting graph $G_2$ and define

$$\mathcal{R} := \{P : P \text{ is a maximal path in } (O_1 \Delta O_2) \cap L\}.$$ 

We call the elements of $\mathcal{R}$ requests.

An important consequence of our restriction to the metric space on the line is that $|\mathcal{R}| \leq k$ (in fact, each of the $k$ maximal paths of $O_1 \Delta O_2$ is contained in $L$ after removing its first and last edge).

Lemma 9. $|\mathcal{R}| \leq k$ and each $R \in \mathcal{R}$ starts and ends with an edge of $O_1$.

For simplification of the analysis we make the following assumptions. In Appendix B.9 we show that they are without loss of generality.

Assumption A For all $i \in W_L$ and all $R \in \mathcal{R}$, either $P(i) \cap R \neq \emptyset$ or $P(i) \subseteq R$, or $R \subseteq P(i)$.

Assumption B For all $j \in W_H$, if $\bigcup_{R \in \mathcal{R}} R \cap P_j \neq \emptyset$, then the first and last edge of $P(j)$ are in $\bigcup_{R \in \mathcal{R}} R$.

From the set of requests, we will determine a subset of at most $2k$ edges that we delete from $M_1$. To this end, we assign each request to a light node in $W_L$ as follows. For $R \in \mathcal{R}$ we define $i_R := \max\{i \in W_L : R \subseteq P(i)\}$, i.e., $P(i_R)$ is the inclusionwise minimal interval of a light node containing $R$. For $i \in W_L$, let

$$\mathcal{R}(i) := \{R \in \mathcal{R} : i_R = i\}.$$ 

Furthermore, we also keep track of the gaps between the requests in $\mathcal{R}(i)$ as follows. For $i \in W_L$, let

$$\tilde{\mathcal{R}}(i) := \{\tilde{R} \subseteq P(i) : \tilde{R} \text{ is a maximal path in } P(i) \setminus \bigcup_{R \in \mathcal{R}(i)} R \text{ and } \tilde{R} \subseteq P(j) \text{ for some } j \in \operatorname{ch}(i)\}.$$ 

For notational convenience we also define $\tilde{\mathcal{R}} := \bigcup_{i \in W_L} \tilde{\mathcal{R}}(i)$. Note that $R' \cap R'' = \emptyset$ for all $R', R'' \in \mathcal{R}(i) \cup \tilde{\mathcal{R}}(i)$. However, $\tilde{R} \in \tilde{\mathcal{R}}(i)$ may contain a request $R \in \mathcal{R}(j)$ from descendants $j$ of $i$. See Fig. 4(c) for an illustration of the assignment.

For $i \in W_L$, let $W_H(i) := \operatorname{ch}(i)$ and $W_L(i) := \{i' \in W : i' \in \operatorname{ch}(j) \text{ for some } j \in W_H(i)\}$. Note that $W_H(i) \subseteq W_H$ and $W_L(i) \subseteq W_L$. Before we can state the algorithm for computing the second stage reply, we need one final lemma.

Lemma 10. Let $i \in W_L$. For every $R \in \mathcal{R}(i)$, there is a $j \in W_H(i)$ with $P(j) \cap R \neq \emptyset$. For every $\tilde{R} \in \tilde{\mathcal{R}}(i)$, there is an $i' \in W_L(i)$ with $P(i') \cap R \neq \emptyset$. 

Algorithm 2.2 works as follows: (i) Create the matching \( M' \) by removing the following edges from \( M_1 \) for each \( i \in W_L \):

1. The edge \( e(i) \) if \( i \neq 0 \) and \( R(i) \neq \emptyset \).
2. For each \( R \in R(i) \) the edge \( e(j_R) \) where \( j_R := \min \{ j \in W_H(i) : P(j) \cap R \neq \emptyset \} \).
3. For each \( \bar{R} \in \mathcal{R}(i) \) the edge \( e(i_{\bar{R}}) \) where \( i_{\bar{R}} := \min \{ i' \in W_L(i) : P(i') \cap \bar{R} \neq \emptyset \} \).

(ii) Let \( M'' \) be a min-cost matching on all vertices not covered by \( M' \) in \( G_2 \). Return \( M_2 := M' \cup M'' \).

Let \( Z \) be indices of the edges removed in step (i). It is not hard to see that \( |R(i)| \leq |\mathcal{R}(i)| \) for each \( i \in W_L \) and therefore \( |Z| \leq 2k \), bounding the recourse of Algorithm 2.2 as intended.

Lemma 11. \( |Z| \leq 2k \).

Now let \( Y := W \setminus (Z \cup \{0\}) \) the nodes corresponding to edges that have not been removed and

\[ \bar{Y} := \{ i \in Y : T[i] \setminus \{0, i\} \subseteq Z \} \]

the nodes that correspond to maximal intervals that have not been removed.

The following lemma is a consequence of the exponential decay property. It shows that in order to establish a bound on the cost of \( M_2 \), it is enough to bound the cost of all paths \( P(i) \) for \( i \in \bar{Y} \).

Lemma 12. \( c(M_2) \leq c(O_2) + 3 \sum_{i \in \bar{Y}} c(P(i)) \)

It remains to bound the cost of the paths associated with the tree nodes in \( \bar{Y} \). We establish a charging scheme by partitioning the line into three areas \( A, B, C \):

1. For \( R \in \mathcal{R} \), let \( A(R) := R \setminus P(j_R) \). We define \( A := \bigcup_{R \in \mathcal{R}} A(R) \).
2. For \( i \in W_L \) and \( \bar{R} \in \mathcal{R}(i) \), let \( B(\bar{R}) := \bar{R} \setminus \bigcup_{P \in W_L(i) \cap Z} P \).
   We define \( B := \bigcup_{\bar{R} \in \bar{R}} B(\bar{R}) \).
3. We define \( C := L \setminus (A \cup B) \).

Consider a set \( A(R) \) for some \( R \in \mathcal{R} \). Recall that \( i_R \) is the index of the smallest light interval constructed by Algorithm 2.1 containing \( R \) and that \( j_R \) is the first child interval of \( i_R \) created by Algorithm 2.1 that intersects \( R \). From the choice of \( j_R \) and the greedy construction of \( P(j_R) \) as a path of maximum \( X(j_R) \), gain we can conclude that \( A(R) \) is not \( O_1 \)-heavy; see Fig. 4 for an illustration. Therefore \( c(A(R) \setminus O_1) > \frac{1}{2} c(A(R)) \). Note that \( O_2 \cap A(R) = A(R) \setminus O_1 \), because \( A(R) \subseteq R \). Hence we obtain the following lemma.

Lemma 13. Let \( R \in \mathcal{R} \). Then \( \frac{1}{2} c(A(R)) \leq c(O_2 \cap A(R)) \).

A similar argument implies the same bound for all sets of the type \( B(\bar{R}) \) for some \( \bar{R} \in \mathcal{R}(i) \) and \( i \in W_L \).

Lemma 14. Let \( \bar{R} \in \mathcal{R} \). Then \( c(O_2 \cap B(\bar{R})) \geq \frac{1}{3} c(B(\bar{R})) \).
Maintaining Perfect Matchings at Low Cost

Fig. 5. Illustration of the proof of Lemma 13. The solid dark grey lines depict edges in $O_1 \cap R$. At the time when Algorithm 2.1 constructed $P^{(j_R)}$, no other child interval of $i_R$ intersecting with $R$ was present. Thus $X^{(i_R)} \cap R = X^{(i_R+1)} \cap R = O_1 \cap R$. If $A(R)$ was $O_1$-heavy, then $P^{(j_R)} \cup A(R)$ would be an $O_1$-heavy path of higher $O_1$-gain than $P^{(j_R)}$, contradicting the greedy construction.

Furthermore, one can show that the sets of the form $A(R)$ and $B(\bar{R})$ and the set $C$ form a partition of $L$. We define $x : L \to \mathbb{R}_+$ by

$$x(e) := \begin{cases} 
\frac{1}{3} & \text{if } e \in A \cup B \\
1 & \text{if } e \in C \cap O_2 \\
0 & \text{if } e \in C \setminus O_2
\end{cases}$$

and obtain the following lemma.

**Lemma 15.** $\sum_{e \in L} c(e)x(e) \leq c(O_2 \cap L)$.

We are now able to bound the cost of each path $P^{(i)}$ for $i \in \bar{Y}$ against its local budget $\sum_{e \in P^{(i)}} c(e)x(e)$.

**Lemma 16.** Let $j \in \bar{Y} \cap W_H$. Then $c(P^{(j)}) \leq 6 \sum_{e \in P^{(j)}} c(e)x(e)$.

For intuition, we give a short sketch for the proof of Lemma 16. Note that $c(P^{(j)}) \leq \frac{3}{2}c(P^{(j)} \cap O_1)$ because $P^{(j)}$ is $O_1$-heavy. Hence it suffices to show that a significant portion of $P^{(j)} \cap O_1$ is contained in the support of $x$. Consider an edge $e \in P^{(j)} \cap O_1$. If $e \notin O_2$ then there must be a request $R \in \mathcal{R}$ with $e \in R$. Note that $j \neq j_R$, since $j$ was not removed Algorithm 2.2. Thus, $e \in A(R) = R \setminus P^{(j_R)}$ or $j_R$ is a descendant or ancestor of $j$. The complete proof, given in Appendix B.7, establishes that $j_R$ cannot be an ancestor of $j$ (under Assumption A) and bounds the total cost of $O_1$-edges contained in child intervals of $j$.

**Lemma 17.** Let $i \in \bar{Y} \cap W_L$. Then $c(P^{(i)}) \leq 3 \sum_{e \in P^{(i)}} c(e)x(e)$.

Because the paths $P^{(i)}$ for $i \in \bar{Y}$ are pairwise disjoint, Lemmas 15 to 17 imply $\sum_{i \in \bar{Y}} c(P^{(i)}) \leq 6c(O_2)$. Plugging this into Lemma 12 completes the proof of Theorem 2:

$$c(M_2) \leq c(O_2) + 3 \sum_{i \in \bar{Y}} c(P^{(i)}) \leq 19c(O_2).$$
A Omitted proofs from Section 3

For convenience, we define the projection \( \psi(e) := L[v, w] \) that maps an edge \( e = \{v, w\} \in E(G_1) \) to the corresponding subpath \( L[v, w] \). We first establish some useful helper lemmas.

**Lemma A1.** Let \( X \subseteq E(G_1) \) and let \( A, B \subseteq E(G_1) \) be two \( X \)-heavy (\( X \)-light, respectively) sets with \( A \cap B = \emptyset \). Then \( A \cup B \) is \( X \)-heavy (\( X \)-light, respectively).

**Proof.** Let \( X \subseteq E(G_1) \) and \( A, B \subseteq E(G_1) \) be \( X \)-heavy sets with \( A \cap B = \emptyset \). It follows that

\[
c((A \cup B) \cap X) = c(A \cap X) + c(B \cap X) \\
\geq 2 \cdot c(A \setminus X) + 2 \cdot c(B \setminus X) = 2 \cdot c((A \cup B) \setminus X).
\]

The proof for two \( X \)-light sets follows analogously. \( \square \)

**Lemma A2.** Let \( X \subseteq E(G_1) \) and let \( A, B \subseteq E(G_1) \) with \( B \subseteq A \). If \( A \) is \( X \)-heavy and \( \text{gain}_X(B) < 0 \), then \( A \setminus B \) is \( X \)-heavy. If \( A \) is \( X \)-light and \( \text{gain}_X(B) > 0 \), then \( A \setminus B \) is \( X \)-light.

**Proof.** Let \( X \subseteq E(G_1) \) and \( A, B \subseteq E(G_1) \) such that \( B \subseteq A \), \( A \) is \( X \)-heavy and \( \text{gain}_X(B) < 0 \). Assume for contradiction that \( A \setminus B \) is not \( X \)-heavy, that is, \( c((A \setminus B) \cap X) < 2 \cdot c((A \setminus B) \setminus X) \). We derive a contradiction to \( A \) being \( X \)-heavy, that is,

\[
c(A \cap X) = c((A \setminus B) \cap X) + c(B \cap X) < c((A \setminus B) \cap X) + c(B \setminus X) \\
< 2c((A \setminus B) \setminus X) + c(B \setminus X) \leq 2c(A \setminus X).
\]

The first inequality follows from \( \text{gain}_X(B) < 0 \) and the second from the assumption that \( A \setminus B \) is not heavy. The proof of the second part of the lemma proceeds analogously. \( \square \)

A.1 Proof of Lemma 3

**Lemma 3.** \( O_1 \) is the unique perfect matching contained in \( L \).

**Proof.** Enumerate the edges of \( O_1 \) by \( e_1, \ldots, e_n \) in arbitrary order. Then let \( Q := \psi(e_1) \Delta \ldots \Delta \psi(e_n) \), i.e., the symmetric difference of the projections of the edges to the line. Note that \( Q \subseteq L \) by construction and that \( Q \) is a \( V(G_1) \)-join because \( \psi(\{v, w\}) \) is a \( \{v, w\} \)-join for each edge \( \{v, w\} \in O_1 \). Because \( L \) is a path, it contains only a unique \( V(G_1) \)-join, namely the matching consisting of every odd edge of \( L \). Furthermore \( Q \subseteq \bigcup_{i=1}^n \psi(e_i) \). Hence \( c(Q) \leq \sum_{i=1}^n c(\psi(e_i)) = c(O_1) \), with strict inequality if \( \psi(e_i) \cap \psi(e_j) \neq \emptyset \) for some \( i \neq j \). We conclude that \( Q = O_1 \). \( \square \)
A.2 Proof of Lemma 5

The following lemma will be useful in the proof of Lemma 5.

**Lemma A3.** Let \( X \subseteq L \) be a matching.

1. Let \( P \) be an \( X \)-heavy \( X \)-alternating path maximizing \( \text{gain}_X(P) \). If \( X \) covers all vertices in \( V(\psi(P)) \), then \( P = \psi(P) \).
2. Let \( I \subseteq L \) be an interval. Then there is an \( X \)-alternating path \( P \) such that \( c(P \cap X) = c(X \cap I) \) and \( c(P \setminus X) = c(I \setminus X) \).

**Proof.** 1. Assume for contradiction that \( X \subseteq L \), \( X \) covers all vertices in \( V(\psi(P)) \) but \( P \neq \psi(P) \). We show that \( \psi(P) \) is then a \( X \)-alternating \( X \)-heavy path with higher gain than \( P \). Note that \( \psi(P) \) is \( X \)-alternating since \( X \) covers all vertices in \( V(\psi(P)) \). Moreover, it holds that \( c(P \cap X) \leq c(\psi(P) \cap X) \) since \( X \subseteq L \) and it holds that \( c(P \setminus X) > c(\psi(P) \setminus X) \). Hence, \( \psi(P) \) is \( X \)-heavy and \( \text{gain}_X(P) < \text{gain}_X(\psi(P)) \), a contradiction to the maximality of \( P \).

2. Let \( X \subseteq L \) be a matching and \( I \subseteq L \) be an interval. Consider \( P := (I \cap X) \cup \{e(P') : P' \text{ maximal path in } I \setminus X\} \). Then, \( P \) is \( X \)-alternating, \( c(P \cap X) = c(I \cap X) \) and \( c(P \setminus X) = c(I \setminus X) \). \( P \) is then a \( X \)-alternating since \( \psi(P) \) covers all vertices in \( V(\psi(P)) \). Note that minimality of \( \psi(P) \) and \( \text{gain}_X(P') = \text{gain}_X(P) - \text{gain}_X(Q) > \text{gain}_X(P) \). This yields a contradiction to the gain-maximality of \( P \).

For the next statement we define a prefix of a path \( P \subseteq E(G_1) \) as a non-empty subset \( Q \subseteq P \) such that \( Q \) is a path and \( P \setminus Q \) is a path.

**Lemma A4.** Let \( X \subseteq E(G_1) \) be a matching and \( P \subseteq E(G_1) \) be a \( X \)-heavy, \( X \)-alternating path that is a maximizer of \( \text{gain}_X(P) \). Let \( Q \) be a prefix of \( P \). Then, \( \text{gain}_X(Q) \geq 0 \).

**Proof.** Assume by contradiction that \( \text{gain}_X(Q) < 0 \). Since \( Q \) is a prefix of the \( X \)-alternating path \( P \), \( P' := P \setminus Q \) is a \( X \)-alternating path. Moreover, with Lemma A2 we know that \( P' \) is \( X \)-heavy. Lastly, we obtain \( \text{gain}_X(P') = \text{gain}_X(P) - \text{gain}_X(Q) > \text{gain}_X(P) \). This yields a contradiction to the gain-maximality of \( P \).

**Lemma 5.** 1. \( X^{(i)}, P^{(i)} \subseteq L \) for all \( i \in [n] \).
2. For all \( i, j \in [n] \) with \( i < j \), either \( P^{(i)} \cap P^{(j)} = \emptyset \) or \( P^{(j)} \subseteq P^{(i)} \).

**Proof.** We say a pair \((i, j)\) with \( i < j \) is violating if \( \psi(P^{(i)}) \cap \psi(P^{(j)}) \neq \emptyset \) and \( \psi(P^{(j)}) \setminus \psi(P^{(i)}) \neq \emptyset \). We will show that no violating pair exists. This proves the lemma as the following claim asserts.

**Claim.** If \( P^{(j)} \neq \psi(P^{(j)}) \), then there is a violating pair \((i', j')\) with \( i' < j' \leq j \).

**Proof.** Let \( j' \) be minimal with \( P^{(j')} \neq \psi(P^{(j')}) \). Note that minimality of \( j' \) implies that \( X^{(j')} = O_1 \Delta P^{(1)} \Delta \ldots \Delta P^{(j'-1)} \subseteq L \). Then Lemma A3 implies that there must be a vertex \( v \in V(\psi(P^{(j')})) \) not covered by \( X^{(j')} \). Because \( X^{(j')} \) covers exactly those vertices not covered by \( \{e(i') : i' < j'\} \), there must be an \( i' < j' \) such that \( v \) is an endpoint of \( P^{(i')} \). The vertex \( v \) cannot be an endpoint of \( P^{(j')} \), because \( v \) is exposed in \( X^{(j')} \) and \( P^{(j')} \) starts and ends with edges from \( X^{(j')} \). This implies that \((i', j')\) is a violating pair. \( \square \)
Now let us assume there are no violating pairs. Then \( P^{(i)} = \psi(P^{(i)}) \subseteq L \) for all \( i \in [n] \) by the claim, which also implies \( X^{(i)} \subseteq L \). This implies the lemma as, in this situation, the condition for violating pairs coincides with the condition in point 2 of the lemma.

By contradiction assume there is a violating pair. Choose \( j \) such that \( j \) is minimal among all possible choices of violating pairs. Then choose \( i \) such that it is maximal for that \( j \) among all violating pairs.

Note that the claim implies that \( P^{(i)}, X^{(i)}, X^{(j)} \subseteq L \). Furthermore, our choice of \( i \) and \( j \) implies that \( \psi(P^{(j)}) \cap \psi(P^{(j)}) = \emptyset \) for all \( j' \) with \( i < j' < j \), as otherwise \((i, j')\) or \((j', j)\) would be a violating pair. In particular, \( P^{(j')} \cap P^{(j)} = \emptyset \) for all \( j' \) with \( i < j' < j \) and thus

\[
X^{(j)} \cap P^{(j)} = X^{(j+1)} \cap P^{(j)} = (X^{(i)} \Delta P^{(i)}) \cap P^{(j)}.
\]

Now consider \( I_1 := \psi(P^{(j)}) \cap P^{(i)} \) and \( I_2 := \psi(P^{(j)}) \setminus P^{(i)} \), both of which are non-empty since \((i, j)\) is a violating pair. Then \( I_1 \) implies \( X^{(j)} \cap I_1 = I_1 \setminus X^{(i)} \) and \( I_1 \setminus X^{(j)} = X^{(i)} \cap I_1 \). With Lemma A3 we conclude that \( \text{gain}_{X^{(i)}}(I_1) = -\text{gain}_{X^{(i)}}(I_1) \leq 0 \), as \( I_1 \) is a prefix or of the gain-maximizing path \( P^{(i)} \).

Therefore \( I_2 = \psi(P^{(j)}) \setminus I_1 \) is \( X^{(i)} \)-heavy by Lemma A2. But then \( I_2 \) is also \( X^{(i)} \)-heavy because \( I_1 \) implies \( X^{(j)} \cap I_2 = X^{(i)} \cap I_2 \). Hence \( I' := P^{(i)} \cup I_2 \) is \( X^{(i)} \)-heavy by Lemma A3 and further

\[
\text{gain}_{X^{(i)}}(I') = \text{gain}_{X^{(i)}}(P^{(i)}) + \text{gain}_{X^{(i)}}(I_2) > \text{gain}_{X^{(i)}}(P^{(i)}),
\]

because \( \text{gain}_{X^{(i)}}(I_2) \geq \frac{1}{2}c(I_2) \). By Lemma A3 there is an \( X^{(i)} \)-heavy, \( X^{(i)} \)-alternating path with higher gain than \( P^{(i)} \), a contradiction. \( \square \)

B. Omitted Proofs from Section 4

B.1 Proof of Lemma 9

**Lemma 9.** \( |\mathcal{R}| \leq k \) and each \( R \in \mathcal{R} \) starts and ends with an edge of \( O_1 \).

**Proof.** We first show that for every \( \{u, v\} \in O_2 \), either \( \{u, v\} \in L \) or \( \{u, v\} \notin V(G_1) \). By contradiction assume there is an edge \( \{u, v\} \in O_2 \setminus L \) but \( u, v \in V(G_1) \). Because \( \{u, v\} \notin L \), there is \( v' \in V(\psi(\{u, v\}) \setminus \{u, v\}) \). Let \( v' \) be the matching partner of \( v' \) in \( O_2 \). Let \( v_1, \ldots, v_4 \) be an ordering of \( \{u, v, u', v'\} \) such that \( v_1 < \cdots < v_4 \). Note that \( \{u, v'\} \cap \{u', v'\} \neq \emptyset \) and hence \( c(u, v) + c(u', v') > c(v_1, v_2) + c(v_3, v_4) \). Thus \( O_2 \setminus \{\{u, v\}, \{u', v'\}\} \cup \{\{v_1, v_2\}, \{v_3, v_4\}\} \) is a matching of lower cost than \( O_2 \), a contradiction.

Now let \( P \) be a connected component of \( O_1 \Delta O_2 \). Note that \( P \) cannot be a cycle, because then \( V(P) \subseteq V(G_1) \) but \( P \not\subseteq L \), contradicting the observation above. Thus \( P \) is a path starting and ending with an edge of \( O_2 \). Because it is \( O_1 \)-alternating, every internal vertex of \( P \) is in \( V(G_1) \). Hence \( P \cap L \) contains all of \( P \) except for its first and last edge. Therefore, \( P \cap L \) is the only request intersecting with \( P \) and it starts and ends with an edge of \( O_1 \). This proves the lemma, because every request has to intersect with a connected component of \( O_1 \Delta O_2 \) and there are only \( k \) of them. \( \square \)
B.2 Proof of Lemma 10

Lemma 10. Let \( i \in W_L \). For every \( R \in \mathcal{R}(i) \), there is a \( j \in W_H(i) \) with \( P^{(j)} \cap R \neq \emptyset \). For every \( \bar{R} \in \mathcal{R}(i) \), there is an \( i' \in W_L(i) \) with \( P^{(i')} \cap R \neq \emptyset \).

Proof. Let \( R \in \mathcal{R}(i) \). By Lemma 9, there must be an edge \( e \in R \cap O_1 \). Because \( P^{(i)} \cap X^{(i)} = P^{(i)} \setminus O_1 \) by Lemma 6, the path \( P^{(i)} \) starts and ends with an edge of \( L \setminus O_1 \). Therefore \( P^{(i)} \cap O_1 \subseteq X^{(i\!+\!1)} \) and hence \( P^{(i)} \cap O_1 \subseteq \bigcup_{j \in \text{ch}(i)} P^{(j)} \) by construction through Algorithm 2.1. Hence \( e \in P^{(j)} \cap \mathcal{R}(i) \) for some \( j \in \text{ch}(i) \subseteq W_H(i) \).

Let \( \bar{R} \in \mathcal{R}(i) \). Since \( \bar{R} \) is a maximal path in \( P^{(i)} \setminus \bigcup_{R \in \mathcal{R}(i)} R \) and not a prefix of \( P^{(i)} \), there are two requests \( R', R'' \in \mathcal{R}(i) \) neighboring \( \bar{R} \) (i.e., \( R' \) and \( R'' \) each have exactly one endpoint with \( \bar{R} \) in common). By Lemma 9, \( R' \) and \( R'' \) both start and end with edges of \( O_1 \). Therefore \( \bar{R} \) starts and ends with edges in \( L \setminus O_1 \), so in particular there is an \( e \in \bar{R} \cap (L \setminus O_1) \). Let \( j \in \text{ch}(i) \) be the unique child such that \( \bar{R} \subseteq P^{(j)} \). Analogously to the argument for \( P^{(i)} \cap O_1 \) given above, \( P^{(j)} \setminus O_1 \subseteq \bigcup_{e \in \text{ch}(j)} P^{(e)} \). Hence \( e \in P^{(j')} \) for some \( j' \in \text{ch}(j) \subseteq W_L(i) \).

B.3 Proof of Lemma 11

Lemma B1. Let \( i \in W_L \). If \( \mathcal{R}(i) \neq \emptyset \) then \( |\mathcal{R}(i)| \leq |\mathcal{R}(i)| - 1 \).

Proof. Note that there are exactly \( |\mathcal{R}(i)| + 1 \) maximal paths in \( P^{(i)} \setminus \bigcup_{R \in \mathcal{R}(i)} R \). This is because requests \( R \in \mathcal{R}(i) \) start and end with edges from \( O_1 \) and are in particular not prefixes of the interval \( P^{(i)} \). Hence, there are two maximal paths in \( P^{(i)} \setminus \bigcup_{R \in \mathcal{R}(i)} R \) that are prefixes of \( P^{(i)} \). In particular these cannot be subsets of \( P^{(j)} \) for any \( j \in \text{ch}(i) \). Summarizing, there are at most \( |\mathcal{R}(i)| - 1 \) intervals that fulfill both conditions in the definition of \( \mathcal{R}(i) \).

Lemma 11. \( |Z| \leq 2k \).

Proof. Let \( Z^{(i)} \) be the set of edge indices tagged for removal when processing node \( i \in L \). We show \( |Z^{(i)}| \leq 2|\mathcal{R}(i)| \), proving the lemma. If \( \mathcal{R}(i) = \emptyset \), then \( Z^{(i)} = \emptyset \). If \( \mathcal{R}(i) \neq \emptyset \), then \( |Z^{(i)}| \leq |\mathcal{R}(i)| + |\mathcal{R}(i)| + 1 \leq 2|\mathcal{R}(i)| \), where the last inequality is due to Lemma 11.

B.4 Proof of Lemma 12

For proving the following lemmas, we formalize the notion of the set of descendants of a tree-node \( i \in W \) by defining \( \text{desc}(i) := \{ j \in W : i \in V(T[j]), j \neq i \} \).

Lemma 12. \( c(M_2) \leq c(O_2) + 3 \sum_{i \in Y} c(P^{(i)}) \)

Proof. Recall that \( M_2 = M' \cup M'' \), where \( M' = \{ e^{(i)} : i \in Y \} \) and \( M'' \) is a minimum cost perfect matching on the set of vertices \( U := \{ v \in V(G_2) : v \) is not covered by \( M' \} \).
Lemma 13. Let $i_1, \ldots, i_\ell$ be an arbitrary ordering of the indices in $Y$ and consider the symmetric difference of paths $P^{(i)}$ for $i \in Y$, i.e., $\bar{M} := P^{(i_1)} \Delta \ldots \Delta P^{(i_\ell)}$. Note that $\bar{M}$ is a $(V(G_2) \setminus U)$-join (as each path in $P^{(i)}$ for $i \in Y$ corresponds to an edge in $\bar{M}'$) and that $\bar{M} \subseteq \bigcup_{i \in Y} P^{(i)}$ by construction of $\bar{Y}$. Thus $c(\bar{M}) \leq \sum_{i \in Y} c(P^{(i)}).

Because $O_2$ is a $V(G_2)$-join, $\bar{M} \Delta O_2$ is a $U$-join. Therefore $c(M'') \leq c(\bar{M} \Delta O_2) \leq c(\bar{M}) + c(O_2)$. Furthermore,

$$c(M') = \sum_{i \in Y} c(P^{(i)}) \leq \sum_{i \in Y} \left( c(P^{(i)}) + \sum_{j \in \text{desc}(i)} c(P^{(j)}) \right) \leq 2 \sum_{i \in Y} c(P^{(i)}).$$

The last inequality follows from the exponential decay property established in Lemma 7. Putting this together, we obtain $c(M_2) = c(M') + c(M'') \leq c(O_2) + c(M') + c(\bar{M}) \leq c(O_2) + 3 \sum_{i \in Y} c(P^{(i)}).$ 

\[\square\]

### B.5 Proofs of Lemmas 13 and 14

**Lemma B2.** Let $i \in W$ and $R \subseteq P^{(i)}$ be an interval. Let $Q := R \setminus P^{(j^*)}$ where $j^* := \min \{j \in \text{ch}(i) : P^{(j)} \cap R \neq \emptyset\}$. If $i \in W_L$, then $Q$ is not $O_1$-heavy. If $i \in W_H$, then $Q$ is not $O_1$-light.

**Proof.** By contradiction assume that $i \in W_L$ and $Q$ is $O_1$-heavy. Note that $P^{(j)} \cap R = \emptyset$ for all $j \in \text{ch}(i)$ with $j < j^*$ by choice of $j^*$. This implies that $X^{(j^*)} \cap R = (X^{(i)} \Delta P^{(i)}) \cap R = O_1 \cap R$, since $i \in W_L$. In particular, $Q$ is $X^{(j^*)}$-heavy. Since $Q$ is $X^{(j^*)}$-heavy, $P^{(j^*)} \cup Q$ is an $X^{(j^*)}$-heavy $X^{(j^*)}$-alternating path by Lemma A1. Furthermore, $\text{gain}_{X^{(j^*)}}(Q \cup P^{(j^*)}) = \text{gain}_{X^{(j^*)}}(Q) + \text{gain}_{X^{(j^*)}}(P^{(j^*)})$, contradicting the construction of $P^{(j^*)}$ by Algorithm 2.1. The proof for the case that $i \in W_H$ and $Q$ is $O_1$-light follows analogously. \[\square\]

In the proof of Lemma 14, we make use of the following consequence of Assumptions A and B

**Lemma B3.** Let $i \in W_L$. If $R(i) = \emptyset$, then $R(i') = \emptyset$ for all $i' \in \text{desc}(i) \cap W_L$.

**Proof.** We show the contrapositive, i.e., if $R(i') \neq \emptyset$ for some $i' \in \text{desc}(i) \cap W_L$, then $R(i) \neq \emptyset$. Let $i' \in \text{desc}(i) \cap W_L$ and $R \in R(i')$. Note that because $i' \in W_L$, there must be $j \in \text{ch}(i)$ with $i' \in \text{desc}(j)$. Hence $R \cap P^{(j)} \supseteq R \cap P^{(i')} \neq \emptyset$. By Assumption B3, there is a $R' \in R$ containing the first edge $e$ of $P^{(j)}$. Note that $e \in P^{(j)}$ but $e$ is not contained in any child interval of $j$, because all intervals created by Algorithm 2.1 have disjoint endpoints. This implies that $i_{R'} \leq i$, because $e \in R' \subseteq P^{(i_{R'})}$. If $i_{R'} = i$, then $R(i) \neq \emptyset$ and we are done. If $i_{R'} < i$, then $R' \not\subseteq P^{(i)}$ and hence, by Assumption A, $P^{(i)} \subseteq R'$. But then $P^{(i)} \subseteq R'$, which implies $R(i') = \emptyset$ as request intervals are disjoint, a contradiction. \[\square\]

**Lemma 13.** Let $R \in R$. Then $\frac{1}{2}c(A(R)) \leq c(O_2 \cap A(R))$. 
Proof. Lemma \[\text{(B2)}\] implies that \(A(R) = R \setminus P^{(j^*_R)}\) is not \(O_1\)-heavy. Note that \(A(R) \setminus O_2 = A(R) \cap O_1\) because \(A(R) \subseteq R \subseteq O_1\Delta O_2\). Thus \(c(A(R) \setminus O_2) = c(A(R) \cap O_1) < 2c(A(R) \setminus O_1) = 2c(A(R) \cap O_1).\)

\[\square\]

**Lemma 14.** Let \(\bar{R} \in \bar{\mathcal{R}}\). Then \(c(O_2 \cap B(\bar{R})) \geq \frac{1}{4} c(B(\bar{R})).\)

Proof. Let \(\bar{R} \in \bar{\mathcal{R}}\) and \(i\) be the unique index such that \(\bar{R} \in \mathcal{R}(i)\). Let \(i^*_R := \min\{i' \in W_L(i) : P^{(i')} \cap \bar{R} \neq \emptyset\}\), i.e., the index of the light edge that was removed because of \(\bar{R}\). Then, because of Lemma \[\text{(B2)}\], \(\bar{R} \setminus P^{(i^*_R)}\) is not \(O_1\)-light. We obtain \(B(\bar{R})\) from \(\bar{R} \setminus P^{(i^*_R)}\) by deleting edges from \(\bigcup_{i' \in W_L(i) \cap Z} P^{(i')}\) that intersect with \(\bar{R}\). By Assumption \[\text{(A)}\] we know that \((\bar{R} \setminus P^{(i^*_R)}) \cap \bigcup_{i' \in W_L(i) \cap Z} P^{(i')})\) is the union of \(O_1\)-light intervals and hence \(O_1\)-light (Lemma \[\text{(A1)}\]). We conclude with Lemma \[\text{(A2)}\] that \(B(\bar{R})\) is not \(O_1\)-light. Note that by construction \(B(\bar{R})\) does not intersect with the request of any descendant of \(i\). We obtain that \(O_1 \cap B(\bar{R}) = O_2 \cap B(\bar{R})\) and from \(B(\bar{R})\) being not \(O_1\)-light we get that \(B(\bar{R})\) is not \(O_2\)-light. This concludes the proof.

\[\square\]

**B.6** Proof of Lemma \[\text{(B5)}\]

**Lemma B.4.** \(L = \bigcup_{R \in \mathcal{R}} A(R) \cup \bigcup_{R \in \bar{\mathcal{R}}} B(\bar{R}) \cup C\)

**Proof.** We show \(A(R) \cap B(\bar{R}) = A(R) \cap A(R') = B(\bar{R}) \cap B(\bar{R}') = \emptyset\) for all \(R, R' \in \mathcal{R}, \bar{R}, \bar{R}' \in \bar{\mathcal{R}}\), then the lemma follows directly from the definition of \(C\).

First, let \(R \in \mathcal{R}\) and \(\bar{R} \in \bar{\mathcal{R}}\). By contradiction assume \(R \cap B(\bar{R}) \neq \emptyset\). Let \(i\) be such that \(\bar{R} \in \mathcal{R}(i)\). Note that \(R \cap B(\bar{R}) \subseteq R \cap P^{(i)}\) and hence \(R\) intersects \(P^{(i)}\).

If \(i_R \in \text{anc}(i)\), then \(R \not\subseteq P^{(i)}\) and hence \(P^{(i)} \subseteq R\) by Assumption \[\text{(A)}\]. This implies \(\mathcal{R}(i) = \emptyset\) and hence \(\bar{\mathcal{R}}(i) = \emptyset\), which contradicts \(\bar{R} \in \mathcal{R}(i)\). If \(j_R = i\), then \(R \cap \bar{R} = \emptyset\) because \(\bar{R} \subseteq P^{(i)} \setminus \bigcup_{R' \in \mathcal{R}(i)} R'\). Thus \(i_R \in \text{desc}(i)\) and more specifically \(i_{R'} \in \text{desc}(j_R)\), where \(j_R\) is the unique child of \(i\) that intersects with \(\bar{R}\). But this implies \(\bar{R} \subseteq P^{(i')}\) for some \(i' \in \text{ch}(j_R)\). Thus \(\mathcal{R}(i') \neq \emptyset\) by Lemma \[\text{(B3)}\] and thus \(B(\bar{R}) \subseteq \bar{R} \setminus P^{(i')}\) by construction of \(B(\bar{R})\).

Second, let \(R, R' \in \mathcal{R}\). Then \(R \cap R' \neq \emptyset\) since requests are disjoint and hence \(A(R) \cap A(R') = \emptyset\).

Finally, let \(\bar{R}, \bar{R}' \in \bar{\mathcal{R}}\) and \(i, i'\) such that \(\bar{R} \in \mathcal{R}(i)\) and \(\bar{R}' \in \mathcal{R}(i')\). It is easy to see that if \(i'\) is not a descendant of \(i\) or vice versa, then \(\bar{R} \cap \bar{R}' = \emptyset\). Hence, assume w.l.o.g. assume that \(i' \in \text{desc}(i)\). Then there exists a child of \(i\), say \(i'' \in \text{ch}(i)\), such that \(P^{(i'')} \subseteq P^{(i')}\). Moreover, from \(\mathcal{R}(i') \neq \emptyset\) we deduce that \(\mathcal{R}(i'') \neq \emptyset\). Therefore, \(B(\bar{R}) \cap P^{(i'')} = \emptyset\) which implies \(B(\bar{R}) \cap B(\bar{R}') = \emptyset\).

\[\square\]

**Lemma 15.** \(\sum_{e \in L} c(e)x(e) \leq c(O_2 \cap L)\).

**Proof.** An implication of Lemma \[\text{(A1)}\] is that \(A, B, C\) are pairwise disjoint and hence

\[
\sum_{e \in O_2} c(e) \geq \sum_{e \in A \cup O_2} c(e) + \sum_{e \in B \cap O_2} c(e) + \sum_{e \in C \cap O_2} c(e).
\]
Moreover, Lemma B4 states that all sets \( A(R) \) for \( R \in \mathcal{R} \) and all sets \( B(\bar{R}) \) for \( \bar{R} \in \bar{\mathcal{R}} \) are pairwise disjoint and therefore Lemmas 13 and 14 imply that 
\[
\sum_{e \in (A \cup B) \cap O_2} c(e) = \frac{1}{2} \sum_{e \in A \cup B} c(e).
\]
The lemma follows from plugging in the definition of \( x \).

\[\square\]

### B.7 Proof of Lemma 16

In the proof of Lemma 16, we make use of the following lemma that is a variant to Lemma 7.

**Lemma B5.** Let \( i \in W_H \). Then 
\[
\sum_{j \in \text{ch}(i)} c(P(j) \cap O_1) \leq \frac{1}{2} \cdot c(P(i) \cap O_1).\]

**Proof.** Let \( i \in W_H \). Then 
\[
\sum_{j \in \text{ch}(i)} c(P(j) \cap O_1) \leq \frac{1}{2} \sum_{j \in \text{ch}(i)} c(P(j) \setminus O_1) \leq \frac{1}{2} c(P(i) \setminus O_1) \leq \frac{1}{4} c(P(i) \cap O_1),
\]
where the first inequality follows from the fact that \( P(j) \) is \( O_1 \)-light for all \( j \in \text{ch}(i) \subseteq W_L \), the second follows from the fact that the intervals \( P(j) \) of the children are disjoint and all contained in \( P(i) \), and the last inequality follows from the fact that \( P(i) \) is \( O_1 \)-heavy.

**Lemma 16.** Let \( j \in \bar{Y} \cap W_H \). Then 
\[
c(P(j)) \leq 6 \sum_{e \in P(j)} c(e) x(e).
\]

**Proof.** We first show that each request intersecting with \( P(j) \) is either contained in a child interval of \( j \) or the edges affected by the request are covered by \( A \).

**Claim.** Let \( R \in \mathcal{R} \) with \( R \cap P(j) \neq \emptyset \). Then \( R \cap P(j) \subseteq A(R) \) or \( R \subseteq P(i') \) for some \( i' \in \text{ch}(j) \).

**Proof.** Assume \( R \nsubseteq P(i') \) for any \( i' \in \text{ch}(j) \). In particular, this implies that \( i_R \notin \text{desc}(j) \). Let \( i \) be the parent node of \( j \) in \( T \). We first exclude the possibility that \( i_R \) is an ancestor of \( i \). Indeed, if this was the case then \( R \nsubseteq P(i) \) and thus by Assumption A, \( P(i) \subseteq R \). But then \( P(i) \) can neither contain other request intervals, nor can it intersect any non-request intervals. Therefore \( \mathcal{R}(i) = \emptyset \) and \( i \neq i_R^* \) for all \( R \in \bigcup_{i' \in W_L} \mathcal{R}(i') \). Therefore, \( i \notin Z \), a contradiction to \( j \in \bar{Y} \).

This implies that \( i_R = i \). Further note that \( j \in \bar{Y} \) implies \( j \neq j_R \) for all \( R \in \mathcal{R} \), as otherwise, \( j \) would have been tagged for removal. We conclude that \( j_R \notin \text{ch}(i) \setminus \{j\} \) and thus \( R \cap P(j) \subseteq R \setminus P(j_R) = A(R) \), as \( P(j) \) and \( P(j_R) \) are disjoint.

Let \( Q := \bigcup_{i' \in \text{ch}(j)} P(i') \). Consider \( e \in P(j) \cap O_1 \). Note that the claim implies that if \( e \notin A \cup Q \), then \( e \in O_2 \). We conclude that \( (P(j) \cap O_1) \setminus Q \subseteq A \cup B \cup (C \cap O_2) \).

Further note that 
\[
c(P(j) \cap O_1) \leq \frac{3}{2} c((P(j) \cap O_1) \setminus Q) \leq 6 \sum_{e \in P(j)} c(e) x(e),
\]
where the first inequality follows from the fact that \( P(j) \) is \( O_1 \)-heavy and the last inequality follows from the fact that \( x(e) \geq 1/3 \) for every \( e \in A \cup B \cup (C \cap O_2) \).

This proves the lemma.

\[\square\]
B.8 Proof of Lemma 17

Lemma 17. Let $i \in \bar{Y} \cap W_L$. Then $c(P(i)) \leq 3 \sum_{e \in P(i)} c(e)x(e)$.

Proof. We distinguish two cases:

Case 1 $(P(i) \cup \bigcup_{R \in \mathcal{R}} R = \emptyset)$: Let $j' \in W_H$ be the parent node of $i$ in $T$ and let $i' \in W_L$ be the parent node of $j'$ in $T$. Note that $i \in \bar{Y}$ implies that $i', j' \in Z$. Since heavy nodes are only tagged for removal when their intervals intersect with requests, we conclude that $P(j') \cap \bigcup_{R \in \mathcal{R}(i')} R \neq \emptyset$. Because $P(i) \subseteq P(j') \setminus \bigcup_{R \in \mathcal{R}(i')} R$, there must be non-request interval $\bar{R} \in \mathcal{R}(i')$ with $P(i) \subseteq \bar{R}$. Then $i \notin Z$ implies $P(i) \subseteq B(\bar{R})$. Thus $P(i) \subseteq B$ implies the statement of the lemma.

Case 2 $(P(i) \cap R \neq \emptyset$ for some $R \in \mathcal{R})$: Note that $i \in Y$ implies $\mathcal{R}(i) = \emptyset$. By Lemma 15, $i_R$ must be an ancestor of $i$ in $T$ and hence $R \not\subseteq P(i)$. Thus $P(i) \subseteq R$ by Assumption A:

- If $j_R$ is not an ancestor of $i$, then $P(i) \subseteq R \setminus P(j_R) = A(R) \subseteq A$ and we are done because $x(e) = 1/3$ for all $e \in A$.
- If $j_R$ is an ancestor of $i$, then $P(i) \subseteq R \cap P(j_R)$. Note that $R \cap P(j_R) \subseteq C$, as $R \cap A(R') = \emptyset$ for all $R' \in \mathcal{R} \setminus \{R\}$ and $R \cap B(R) = \emptyset$ for all $R \in \bigcup_{i' \in W_L} \mathcal{R}(i')$. Thus $P(i) \subseteq C$ in this latter case. Then

$$\sum_{e \in P(i)} c(e)x(e) = c(P(i) \cap O_2) = c(P(i) \setminus O_1) \geq \frac{1}{3}c(P(i)),$$

where the first equality follows from $P(i) \subseteq C$, the second equality follows from $P(i) \subseteq R$ and the final inequality follows from the fact that $P(i)$ is $O_1$-light. \hfill \square

B.9 Assumptions A and B are without loss of generality

If $\mathcal{R}$ violates Assumption A or Assumption B, we construct a modified set of requests as follows:

(i) Let $\mathcal{R}':= \mathcal{R}$. (ii) While Assumption A or Assumption B is violated, apply one of the following rules:

1. For $R \in \mathcal{R}'$ and $i \in W_L$ with $R \setminus P(i) \neq \emptyset$ and $P(i) \setminus R \neq \emptyset$: Replace $R$ by $R':= R \setminus P(i)$ (if $R' = \emptyset$, then remove $R$ entirely).
2. For $j \in W_H$ such that $P(j) \cap \bigcup_{R \in \mathcal{R}'} R \neq \emptyset$ such that there is a prefix of $P(j)$ not contained in $\bigcup_{R \in \mathcal{R}'} R$: Let $Q$ be the maximal prefix of $P(j)$ not contained in $\bigcup_{R \in \mathcal{R}'} R$. Let $R$ be the request with $R \cap P(j) \neq \emptyset$ that shares an endpoint with $Q$. Replace $R$ by $R':= R \cup Q$ (if $R'$ shares an endpoint with some other $R'' \in \mathcal{R}'$, then merge $R'$ and $R''$).

It is easy to see that $|\mathcal{R}'| \leq |\mathcal{R}|$ and that $R_1 \cap R_2 \neq \emptyset$ for all $R_1, R_2 \in \mathcal{R}'$. We thus obtain a new set of requests $\mathcal{R}'$ fulfilling Assumptions A and B. We let $O' := O_1 \Delta \bigcup_{R \in \mathcal{R}'} R$. 


Lemma B6. \( c(O') \leq c(O_2 \cap L) \).

**Proof.** Consider an application of Rule 1 to \( i \in W_L \) and \( R \in \mathcal{R} \). Let \( \bar{R} := R \cap P^{(i)} \). Observe that \( \bar{R} \) is a prefix of \( P^{(i)} \) and therefore \( \text{gain}_{X^{(i)}}(\bar{R}) > 0 \). Note that \( X^{(i)} \cap P^{(i)} = P^{(i)} \setminus O_1 \) because \( i \in W_L \). We conclude \( c(\bar{R} \cap O') = c(\bar{R} \cap O_1) < c(\bar{R} \cap O_1) = c(\bar{R} \cap O_2) \).

Consider an application of Rule 2 to \( j \in W_H \) and \( R \in \mathcal{R} \). Note that \( \text{gain}_{X^{(j)}}(Q) > 0 \) because \( Q \) is a prefix of \( P^{(j)} \). Note further that \( X^{(j)} \cap P^{(j)} = P^{(j)} \cap O_1 \) because \( j \in W_L \). We conclude \( c(Q \cap O') = c(Q \setminus O_1) > c(Q \cap O_1) = c(Q \cap O_2) \).

We run Algorithm 2.2 for this modified set of requests, i.e., we use \( \mathcal{R}' \) instead of \( \mathcal{R} \) to determine the matching \( M' \).

As before, let \( Z \) be the set of indices of the removed edges, let \( Y := W \setminus (Z \cup \{0\}) \), and \( \bar{Y} := \{i \in Y : T[i] \setminus \{0, i\} \subseteq Z\} \). Then \( c(M_2) \leq c(O_2) + 3 \sum_{i \in \bar{Y}} c(P^{(i)}) \) by Lemma B6 (note that the proof of the lemma does not make any assumptions about \( Z \)). Furthermore, note that the proofs of Lemmas B9 to B14 work without alteration when replacing \( \mathcal{R} \) by \( \mathcal{R}' \) and \( O_2 \) by \( O' \). We thus obtain \( \sum_{i \in \bar{Y}} c(P^{(i)}) \leq 6c(O') \leq 6c(O_2 \cap L) \) and hence \( c(M_2) \leq 19c(O_2) \).

References

1. M. Ball, L. Bodin, and R. Dial. A Matching Based Heuristic for Scheduling Mass Transit Crews and Vehicles. *Transportation Science*, 17:4–31, 1983.
2. C. E. Bell. Weighted matching with vertex weights: An application to scheduling training sessions in NASA space shuttle cockpit simulators. *European Journal of Operational Research*, 73:443–449, 1994.
3. A. Bernstein, J. Holm, and E. Rotenberg. Online bipartite matching with amortized \( O(\log^2 n) \) replacements. In *Proceedings of the Twenty-ninth Annual ACM Symposium on Discrete Algorithms (SODA ’18)*, pages 947–959, 2018.
4. K. Chaudhuri, C. Daskalakis, R. D. Kleinberg, and H. Lin. Online bipartite perfect matching with augmentations. In *Proceedings of the Twenty-eighth IEEE Conference on Computer Communications (INFOCOM ’09)*, pages 1044–1052, 2009.
5. N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.
6. U. Derigs and A. Metz. A matching-based approach for solving a delivery/pick-up vehicle routing problem with time constraints. *Operations-Research-Spektrum*, 14:91–106, 1992.
7. M. C. Dourado, D. Meierling, L. D. Penso, D. Rautenbach, F. Protti, and A. R. de Almeida. Robust recoverable perfect matchings. *Networks*, 66:210–213, 2015.
8. J. Edmonds. Maximum matching and a polyhedron with \( 0,1 \)-vertices. *Journal of Research of the National Bureau of Standards, Section B*, 69:125–130, 1965.
9. B. Fuchs, W. Hochstättler, and W. Kern. Online matching on a line. *Theoretical Computer Science*, 332:251–264, 2005.
10. A. M. Geoffrion and R. Nauss. Parametric and Postoptimality Analysis in Integer Linear Programming. *Management Science*, 23:453–466, 1977.
11. E. F. Grove, M.-Y. Kao, P. Krishnan, and J. S. Vitter. Online perfect matching and mobile computing. In *Algorithms and Data Structures (WADS ’95)*, volume 955 of *Lecture Notes in Computer Science*, pages 194–205. Springer, 1995.
12. A. Gu, A. Gupta, and A. Kumar. The power of deferral: maintaining a constant-competitive Steiner tree online. In *Proceedings of the Forty-fifth Annual ACM Symposium on Symposium on Theory of Computing (STOC ’13)*, pages 525–534, 2013.
13. A. Gupta and A. Kumar. Online Steiner Tree with Deletions. In *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ’14)*, pages 455–467, 2014.
14. A. Gupta, A. Kumar, and C. Stein. Maintaining assignments online: Matching, scheduling, and flows. In *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ’14)*, pages 468–479, 2014.
15. A. Gupta and K. Lewi. The online metric matching problem for doubling metrics. In *Proceedings of the Thirty-ninth International Colloquium on Automata, Languages and Programming (ICALP ’12)*, pages 424–435, 2012.
16. B. Kalyanasundaram and K. Pruhs. Online weighted matching. *Journal of Algorithms*, 14:478 – 488, 1993.
17. R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing (STOC ’90)*, pages 352–358, 1990.
18. S. Khuller, S. G. Mitchell, and V. V. Vazirani. On-line algorithms for weighted bipartite matching and stable marriages. *Theoretical Computer Science*, 127:255–267, 1994.
19. C. Liebchen, M. Lübbecke, R. Möhring, and S. Stiller. *The Concept of Recoverable Robustness, Linear Programming Recovery, and Railway Applications*, pages 1–27. Springer Berlin Heidelberg, 2009.
20. N. Megow, M. Skutella, J. Verschae, and A. Wiese. The power of recourse for online MST and TSP. *SIAM Journal on Computing*, 45:859–880, 2016.
21. K. Nayyar and S. Raghvendra. An input sensitive online algorithm for the metric bipartite matching problem. In *Proceedings of the Fifty-eighth Annual IEEE Symposium on Foundations of Computer Science (FOCS ’17)*, pages 505–515, 2017.
22. S. Olafsson. Weighted Matching in Chess Tournaments. *The Journal of the Operational Research Society*, 41:17–24, 1990.
23. W. R. Pulleyblank. *Edmonds, Matching and the Birth of Polyhedral Combinatorics*, pages 181–197. Springer Berlin Heidelberg, 2012.
24. S. Raghvendra. A Robust and Optimal Online Algorithm for Minimum Metric Bipartite Matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM ’16)*, volume 60 of *Leibniz International Proceedings in Informatics*, pages 18:1–18:16. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016.
25. S. Raghvendra. Optimal analysis of an online algorithm for the bipartite matching problem on a line. Technical Report arXiv:1803.07206, 2018.
26. E. Reingold and R. Tarjan. On a Greedy Heuristic for Complete Matching. *SIAM Journal on Computing*, 10:676–681, 1981.