Input Sensitive Analysis of a Minimum Metric Bipartite Matching Algorithm

Krati Nayyar

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

Sharath Raghvendra, Chair
Lenwood S. Heath
T. M. Murali

April 28, 2017
Blacksburg, Virginia

Keywords: online algorithms, weighted matching, competitive ratio, input sensitive

Input Sensitive Analysis of a Minimum Metric Bipartite Matching Algorithm

Krati Nayyar

ABSTRACT

In the online metric bipartite matching problem, we are given a set S of server locations in a metric space. Locations of the requests are revealed one at a time and when a request is revealed, we have to immediately and irrevocably match this request to a free server at a cost that is equal to the distance between the server and the request. An α-competitive online algorithm will assign these requests to free servers so that the total cost, for any request sequence R, is at most α times the cost of the minimum matching between S and R. In an adversarial model of request generation, we assume that there is an adversary who has complete knowledge of the online algorithm and its decisions for each request and generates a request sequence that will maximize the competitive ratio α. Note that, at the start, the algorithm has full knowledge of the metric space $\mathbb{M} = (X, d)$ and the server locations S, whereas the request locations R are revealed by an adversary. Therefore, ideally, we would like to pick an algorithm that will achieve the best competitive ratio for this input instance $(\mathbb{M}, S)$ in the adversarial model.

A recently discovered robust deterministic online algorithm (which we refer to as the robust matching or **RM**-Algorithm) works for any arbitrary metric and has been shown to simultaneously achieve a worst-case optimal competitive ratio of $2n - 1$ in the adversarial model and $2H_n - 1$ in the relatively weaker random arrival model. In this thesis, we show that the **RM**-Algorithm in the adversarial model will perform near-optimally with respect to every input instance $(\mathbb{M}, S)$. To help with the analysis, we define the *stretch* of a point set $P \subseteq X$, denoted by $\mu_{\mathbb{M}}(P)$ to be the ratio of the traveling salesman tour and the diameter of P. We also define the *stretch* of a metric space to be the maximum stretch of any set of n points in the metric space $\mathbb{M}$. We will show that, for any input instance $(\mathbb{M}, S)$ and a set $S' \subseteq S$, the **RM**-Algorithm has a competitive ratio of $O(\mu_{\mathbb{M}}(S') \log^2 n)$ in the adversarial model. There is also a straight-forward lower bound of $\Omega(\mu_{\mathbb{M}}(S'))$ on the competitive ratio of any algorithm for the input instance $(\mathbb{M}, S)$, implying that the **RM**-Algorithm is near optimal with respect to each input instance. As consequences, we obtain

the following results for the special case where $\mathbb{M}$ is the d-dimensional Euclidean space:

- Suppose the server locations are points on a line in $\mathbb{R}^d$. Since the stretch of any set of points on a line is 2, the competitive ratio of the **RM**-Algorithm will be $O(\log^2 n)$,
- Suppose the server location set S is a subset of k-dimensional linear subspace in $\mathbb{R}^d$. Since the stretch of any set of n points in a k-dimensional linear subspace of $\mathbb{R}^d$ is $\Theta(n^{1-1/k})$, the competitive ratio of the **RM**-Algorithm will be $O(n^{1-1/k} \log^2 n)$.

For the input sensitive analysis, we replace each request with a small and a large ball. The radius of the small ball is based on the dual weight assigned to each processed request. We express the cost of the online matching as a sum of the radii of the smaller balls. We also relate the cost of the optimal matching to the radius of the larger ball. We then use a novel and simple variant of the well-known Vitali's lemma to partition and relate the radii of these balls to the stretch of the server locations.

Input Sensitive Analysis of a Minimum Metric Bipartite Matching Algorithm

Krati Nayyar

GENERAL AUDIENCE ABSTRACT

In various business and military settings, there is an expectation of on-demand delivery of supplies and services. Typically, several delivery vehicles (also called servers) carry these supplies. Requests arrive one at a time and when a request arrives, a server is assigned to this request at a cost that is proportional to the distance between the server and the request. Bad assignments will not only lead to larger costs but will also create bottlenecks by increasing delivery time. There is, therefore, a need to design decision-making algorithms that produce cost-effective assignments of servers to requests in real-time.

In this thesis, we consider the online bipartite matching problem where each server can serve exactly one request. In the online minimum metric bipartite matching problem, we are provided with a set of server locations in a metric space. Requests arrive one at a time that have to be immediately and irrevocably matched to a *free* server. The total cost of matching all the requests to servers, also known as the *online matching* is the sum of the cost of all the edges in the matching. There are many well-studied models for request generation. We study the problem in the adversarial model where an *adversary* who knows the decisions made by the algorithm generates a request sequence to maximize ratio of the cost of the online matching and the minimum-cost matching (also called the competitive ratio). An algorithm is α-competitive if the cost of online matching is at most α times the minimum cost.

A recently discovered robust and deterministic online algorithm (we refer to this as the robust matching or the **RM**-Algorithm) was shown to have optimal competitive ratios in the adversarial model and a relatively weaker random arrival model. We extend the analysis of the **RM**-Algorithm in the adversarial model and show that the competitive ratio of the algorithm is sensitive to the input, i.e., for "nice" input metric spaces or "nice" server placements, the performance guarantees of the **RM**-Algorithm is significantly better. In fact, we show that the performance is almost optimal for

any fixed metric space and server locations.

To my dear parents...

Acknowledgments

I would first like to extend my sincere gratitude to my thesis advisor, Dr. Sharath Raghvendra for his constant support and guidance all throughout this research work. His expertise and insight in the area have always led this research in the right direction. He always welcomed me for any questions or doubts and has been patient all throughout our discussion meetings. Owing to the brainstorming sessions and discussions with Dr. Raghvendra, I have always appreciated the intricacies of this area and the challenges faced during this work. This thesis would not have been possible without his persistent help and guidance.

I would also like to thank my family for their love and support which has helped me throughout my education and writing of this thesis. It is their belief in me that picks me up in the moment of despair. I would also like to thank my brother and sister for always being there for me and to all my wonderful friends who have always cheered me up in hard times.

Last but not the least, I would like to thank *National Science Foundation (NSF)* for supporting this project financially.

Contents

List of Figures

Chapter 1

Introduction

Driven by consumer demand for quick access to products, business ventures schedule their delivery of goods and services in real-time, often without complete knowledge of future request locations or their order of arrival. Due to this lack of complete information, decisions made tend to be sub-optimal. Also, these decisions must be made immediately and irrevocably. Therefore, there is a need for robust and competitive *online algorithms* that allocate resources to requests in real-time at minimal cost.

These resources are servers placed in various locations S with $|S| = n$ in any arbitrary metric space. Each server has a capacity that restricts how many requests it can serve. When a new request $r \in R$ arrives, one of the servers $s \in S$ that has a positive residual capacity to serve is matched to this request. After this request is served, the capacity of the server reduces by one. The cost associated with this assignment is a metric cost represented by $d(s, r)$; for instance, it could be the minimum distance traveled by the server to reach the request.

The case where the capacity of every server is ∞ is the celebrated k-*server problem*. The case where every server has a capacity of 1 is the *metric bipartite matching problem*. In this case, we are given an input instance of server location S in the metric space $\mathbb{M}$. Requests are generated by an adversary and revealed one at a time. We assume $|S| = |R| = n$. When a request is revealed,

we have to immediately and irrevocably match it to a free server. The resulting assignment is a matching and is referred to as an *online matching*. Finding a minimum-cost matching is impossible as the adversary can easily fill up the remaining request locations in such a way that our current assignment becomes sub-optimal. Therefore, we want our algorithm to compute an online matching M that is a good approximation. The cost of this online matching is given by $w(M)$. Given an input server location set S and metric space $\mathbb{M}$, suppose the adversary chooses a set of requests R and their arrival order. Let M_{OPT} be the minimum-cost matching of S and R. We say that our algorithm is α-*competitive*, for $\alpha \geq 1$, when the cost of the online matching M is at most α times the cost of M_{OPT}, i.e.,

$$w(M) \leq \alpha w(M_{\text{OPT}}).$$

There are several well-studied models for request generation. In the *adversarial model*, there is an adversary who knows the server locations and the assignments made by the algorithm and generates a sequence to maximize α. Another popular model is the *random arrival model* [10] where the adversary chooses the set of request locations R before the algorithm executes but the arrival order is a permutation chosen uniformly at random from the set of all possible permutations of R. In practical situations, it may be useful to assume that the request locations are independently and identically distributed (i.i.d.) from a known or an unknown distribution $\mathcal{D}$. Known and Unknown distribution models are weaker than the random arrival model. This is because the arrival order for any set of n requests generated in these models is a random permutation. Therefore, the competitive ratio of an algorithm in the random arrival model is an upper bound on the competitive ratio in the known and the unknown distribution models; see [6] for an algorithm in these models. Another model of theoretical interest is the *oblivious adversary model*. In this model, the adversary knows the algorithm and decides the request locations and their arrival order. However, the online algorithm is a randomized algorithm and the adversary does not know the random choices made by the algorithm. This model is weaker than the adversarial model but stronger than the random arrival model. It is an open question whether one can design online algorithms for the k-server and the minimum metric bipartite matching problems that simultaneously achieve optimal performance under each of these models. Such algorithms will not only be theoretically superior but may actually be robust

and practically useful for making real time decisions for many business ventures and minimizing the overhead costs.

We would also like to note the role of the input instance, i.e., the metric space from which the request and server locations are chosen and the set of server locations S. The existing algorithms that are designed to work for every input instance $(\mathbb{M}, S)$ are considered optimal if their worst case performance among all possible inputs is minimized. However, in practical scenarios, the input metric and/or the initial server location may be "nice" and may admit online algorithms that produce substantially superior solutions. Therefore, it is desirable to have an algorithm that works optimally for every input instance. In this thesis, we present such an algorithm for the minimum metric bipartite matching problem that simultaneously achieves a near-optimal performance for every input instance $(\mathbb{M}, S)$.

1.1 Preliminaries

In this section, we introduce some of the basic definitions and notations related to this problem.

Consider a metric space $\mathbb{M} = (X, d)$ and let $S, R \subseteq X$. Consider the complete *bipartite graph* $G = (V, E)$ of two disjoint vertex sets S, R where $V = S \cup R$ and $E = S \times R$, see Figure 1.1. A

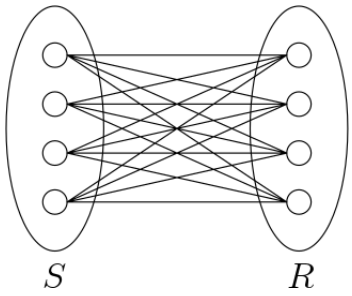

Figure 1.1. Bipartite graph with set of servers S and requests R.

matching $M \subseteq S \times R$ is any set of vertex-disjoint edges of the bipartite graph $G(S, R)$. We denote the cost of server s serving a request r by $d(s, r)$. For any subset $K \subseteq S \times R$, we define the sum of the cost of its individual edges as its *cost* and denote it by $w(K) = \sum_{(s,r)\in K} d(s, r)$. A *perfect*

matching is a matching where every server in S is serving exactly one request in R, i.e., $|M| = n$. A *minimum-cost perfect matching* denoted by M_{OPT} is a perfect matching with the minimum cost which is also known as the *optimal matching*.

1.1.1 Primal and Dual Linear Programs for Minimum-Cost Matching

The minimum-cost matching problem for any bipartite graph $G = (S \cup R, S \times R)$ can be written as a linear program in terms of edge-costs. Let $\delta(v)$ represent the set of edges entering any vertex v.

$$\min \sum_{(u,v)\in G} x_{(u,v)} d(u,v)$$

$$\text{s.t.} \quad x_{(u,v)} = \begin{cases} 1, & \text{if } (u,v) \in M; \\ 0, & \text{otherwise.} \end{cases}$$

$$\sum_{(u,v)\in\delta(v)} x_{(u,v)} = 1; \forall v \in S \cup R$$

$$0 \leq x_{(u,v)}; \forall (u,v) \in S \times R$$

Using Linear Program(LP) Duality, we can write the dual of this LP as follows. We assign dual weights to each of the vertices in the bipartite graph G which is called the *vertex labelling*. The dual weight of any vertex v is represented by $y(v)$ where $y(v)$ is always a real value. For any vertex $s \in S$ and $r \in R$, the edge (s, r) in the bipartite graph should have the *feasible labelling*, given by the following equation:

$$y(s) + y(r) \leq d(s,r).$$

Hence, the dual LP of the minimum-cost matching for bipartite graph G can be written as:

$$\max \sum_{v\in S\cup R} y(v)$$

$$\text{s.t.} \quad y(s) + y(r) \leq d(s,r); \forall e \equiv (s,r); \forall s \in S, \forall r \in R$$

We use a similar dual LP in Chapter 2 for the **RM**-Algorithm.

1.2 Previous Work

Solutions for the k-server problem and the online bipartite matching problem use similar mathematical tools and methodologies. Both of these problems have been extensively studied in the adversarial model and the oblivious model [3, 7, 9, 11]. In the random arrival model, there is an online algorithm for the metric bipartite matching problem that was presented recently [12]. We do not know any existing work for the k-server problem in the random arrival model.

The k-server problem is central to the theory of online algorithms. The problem was first proposed by Manasse *et al.* [11]. In the adversarial model, the best-known deterministic algorithm for this problem is the $(2k-1)$-competitive work function algorithm [9]. It is known that no deterministic algorithm can achieve a competitive ratio better than k. It is conjectured that in fact there is a k-competitive algorithm for this problem. This conjecture is popularly called the *k-server conjecture* [11].

For the online metric bipartite matching problem, in the adversarial model, there is a $(2n-1)$-competitive deterministic algorithm by Khuller *et al.* [7] and by Kalyanasundaram and Pruhs [5]. They also show that there are metric spaces where no online algorithm can achieve a better competitive ratio in the adversarial model. In the special case of the line metric, it is possible to achieve a sub-linear competitive ratio of $O(n^{0.59})$ [1] . Khuller *et al.* [7] point to the possibility of better online algorithms in d-dimensional Euclidean spaces but leave this problem as open.

For an oblivious adversary, there are $\log^{O(1)} n$-competitive algorithms for both the k-server problem and the online metric bipartite matching problem. Bansal *et al.* [3] achieve an $O(\log^2 n)$-competitive algorithm for the metric bipartite matching problem. For the k-server problem, Bansal *et al.* [2] presented an $O(\log^{O(1)} n \log k)$-competitive algorithm. There is also an online algorithm [12] for the metric matching problem that simultaneously achieves optimal competitive ratio of $2n-1$ and $2H_n - 1$ under the adversarial and the random arrival model respectively. Interestingly, the

algorithm relies on a parameter $t > 0$, and the competitive ratio approaches optimality as t tends to ∞. In this paper, we will improve the analysis of this algorithm in the adversarial model.

Note that for a d-dimensional metric space, there is an $O(d \log n)$-competitive algorithm in the oblivious model [4]; here d is the doubling dimension of the metric space. In the adversarial model, the question of finding a deterministic $O(1)$-competitive online algorithm for the line metric is an important open question; see [1, 8] for results on this special case.

1.3 Our Contributions

We begin by describing our results in Section 1.3.1 and our technical contributions in Section 1.3.2.

1.3.1 Our Results

In this thesis, we provide a new analysis of the deterministic online algorithm of [12] (which we refer to as the Robust Matching or **RM**-Algorithm) in the adversarial model. To help with the analysis, we define the *stretch* of a point set $P \subseteq \mathbb{M}$, denoted by $\mu_{\mathbb{M}}(P)$ to be the ratio of the traveling salesman tour and the diameter of P. We also define the *stretch* of a metric space to be the maximum stretch of any set of n points in the metric space $\mathbb{M}$. We will show that, for any input instance $(\mathbb{M}, S)$ and a set $S' \subseteq S$, the **RM**-Algorithm has a competitive ratio of $O(\mu_{\mathbb{M}}(S') \log^2 n)$ in the adversarial model. We also provide a straight-forward lower bound of $\Omega(\mu_{\mathbb{M}}(S'))$ on the competitive ratio of any algorithm for the input instance $(\mathbb{M}, S)$ implying that the **RM**-Algorithm is near optimal with respect to each input instance. It is easy to see that the stretch of the d-dimensional Euclidean space is $\Theta(n^{1-1/d})$ for all $d \geq 1$. Therefore, if $(S \cup R) \subset \mathbb{R}^d$, we will prove a competitive ratio of $O(n^{1-1/d} \log^2 n)$ for the **RM**-Algorithm. We generalize the result to the following:

- Suppose the server locations S are points on a line in $\mathbb{R}^d$ but the adversary is at a liberty to choose the requests R from $\mathbb{R}^d$. Since the stretch of any set of points on a line is 2, the competitive ratio of the **RM**-Algorithm will be $O(\log^2 n)$, and,

- Suppose the server location set S is a subset of k-dimensional linear subspace in $\mathbb{R}^d$ and the adversary chooses requests R from $\mathbb{R}^d$. Since the stretch of any set of n points in a k-dimensional linear subspace of $\mathbb{R}^d$ is $\Theta(n^{1-1/k})$, the competitive ratio of **RM**-Algorithm will be $O(n^{1-1/k}\log^2 n)$.

We also provide the following lower bounds to contrast with our upper bounds:

- There is a lower bound of $\Omega(\mu_{\mathbb{M}}(S'))$ on the performance of any algorithm for the input $(\mathbb{M}, S)$ where $S' = \max_{S''\subseteq S} \mu_{\mathbb{M}}(S'')$. This establishes that the **RM**-Algorithm performs near-optimally under $(\mathbb{M}, S)$.

- There exists an input server configuration S on a line and a request sequence R such that the ratio of the cost of the online algorithm to the cost of the optimal matching is $\Omega(\log n)$. Therefore, it is impossible to achieve an $\Omega(\mu_{\mathbb{M}}(S'))$-competitive analysis of the **RM**-Algorithm.

1.3.2 Technical Contribution

We replace each request with a ball (referred to as an "inner ball") of an appropriate radius and show that the cost of the online matching can be expressed as the sum of the radii of these inner balls. For each request, we also generate a larger ball called the "outer ball" that contains a high density of the edges from the optimal matching. Consider the special case where the inner ball of each request is disjoint from that of any other request and suppose the outer ball covers the request set. In this case, due to the disjointness of inner balls, we can easily bound the sum of the radii of the inner balls by the cost of the traveling salesman tour that visits all the requests. Since outer balls have a high density of edges from the optimal matching, we can also easily relate the optimal matching to the diameter of the request set. However, the radius of the inner balls and outer balls may not satisfy these conditions.

To overcome this challenge, we partition requests into "well-separated" clusters and carefully create an inner ball and outer ball for each such cluster. We then introduce a simple and novel variant of

the well-known Vitali's covering lemma that selects a set of disjoint outer balls that partition the requests into subsets of small diameter . We obtain our result by analyzing each such set separately.

1.4 Thesis Outline

The rest of the thesis is organized as follows; We introduce background, notations, and the **RM**-Algorithm in Chapter 2. We present the input sensitive analysis of the **RM**-Algorithm in Chapter 3. In Chapter 4, we provide the lower bounds on the competitive ratio of the **RM**-Algorithm. We conclude and state future work in Chapter 5.

Chapter 2

Background of the RM-Algorithm

In this chapter, we will reintroduce the relevant definitions, describe the algorithm and present some of its relevant details from [12].

Let S and R be the set of server and request locations. A *matching* $M \subseteq S \times R$ is any set of vertex-disjoint edges of the complete bipartite graph $G(S \cup R, S \times R)$ such that the cost of the matching M is given by $w(M) = \sum_{(s,r)\in M} d(s, r)$.

Given a matching M^* on $G(S \cup R, S \times R)$, an *alternating path* (or cycle) is a simple path (resp. cycle) whose edges alternate between those in M^* and those not in M^*. We refer to any vertex that is not matched in M^* as a *free vertex*. An *alternating tree* is a tree rooted at a free request in which every path is an alternating path. An *augmenting path* P is an alternating path between a free request and a free server. We can *augment* M^* by one edge along P if we remove the edges of $P \cap M^*$ from M^* and add the edges of $P \setminus M^*$ to M^*. After augmenting, the new matching is precisely given by $M^* \oplus P$, where $\oplus$ is the symmetric difference operator.

Notations of the RM-Algorithm For a parameter $t \geq 1$, we define the *t-net-cost* of any augmenting path P to be:

$$\phi_t(P) = t\left(\sum_{(s,r)\in P\setminus M^*} d(s,r)\right) - \sum_{(s,r)\in P\cap M^*} d(s,r).$$

The parameter t is fixed at the beginning of the **RM**-Algorithm. Using these notations, we will now describe the algorithm. It maintains two matchings: an online matching M and an offline matching M^* both of which are initialized to $\emptyset$. After processing $i-1$ requests, the matchings M and M^* will match each of the $i-1$ requests to servers in S such that the set of unmatched servers S_F is the same for both the online matching M and the offline matching M^*. To process the i^{th} request r_i, the algorithm does the following

1. Compute the minimum t-net-cost augmenting path P_i with respect to the offline matching M^*. Let P_i be this path starting from r_i and ending at some server $s_i \in S_F$.
2. Update the offline matching M^* by augmenting it along P_i, i.e., $M^* \leftarrow M^* \oplus P_i$.
3. Update the online matching by matching r_i to s_i. $M \leftarrow M \cup \{(s_i, r_i)\}$.

We refer to the steps taken by the algorithm to process request r_i as phase i of the algorithm. We assume that in Step 1, if the algorithm has multiple minimum t-net-cost augmenting paths, it simply selects the one with the smallest length; where length of an augmenting path is defined as sum of the cost of the edges in the respective path. There is an $O(n^2)$-time primal-dual algorithm to compute such a minimum t-net-cost path in Step 1 of any phase i [12]. This algorithm maintains the dual weights for all servers and requests. These dual weights will play an important role in the metric-sensitive analysis of the algorithm. The offline matching M^* that is maintained by our algorithm is always a *t-feasible matching* which we define next.

For every vertex v in the graph $G(S, R)$, let $y(v)$ represent its dual weight. The offline matching M^* along with the set of dual weights $y(\cdot)$ will be t-feasible if the following conditions hold for

any request $r \in R$ and any server $s \in S$:

$$y(s) + y(r) \leq td(s, r), \tag{2.1}$$

$$y(s) + y(r) = d(s, r) \quad \text{for } (s, r) \in M^*. \tag{2.2}$$

Initially, at the start of phase 1, every request and server will have a dual weight of 0 and the empty matching M^* along with these dual weights forms a t-feasible matching. We assign a dual weight of 0 for those requests of R that have not yet arrived. Also, we refer to an edge $(r, s) \in S \times R$ to be *eligible* if it satisfies the following conditions:

$$y(s) + y(r) = td(s, r), \quad \text{if } (s, r) \notin M^* \tag{2.3}$$

$$y(s) + y(r) = d(s, r) \quad \text{if } (s, r) \in M^*. \tag{2.4}$$

During phase i, we process the request r_i in two sub-phases. The first sub-phase is similar to the Hungarian Search procedure where we compute the shortest t-net-cost path P_i with respect to M^* by growing an *alternating tree* consisting only of eligible edges. In this tree, there is an alternating path from r_i to every server and request participating in this tree. To grow this tree, we adjust the dual weights of every server and request until at least one more edge becomes eligible and a new vertex enters the tree. This search procedure ends when an augmenting path P_i consisting only of eligible edges is found. Let A_i (resp. B_i) be the set of requests (resp. servers) that participated in this alternating tree for request r_i. We would like to note that during this sub-phase, the dual weights of requests in A_i will only increase whereas the dual weights of servers in B_i will only reduce.

The second sub-phase begins once the augmenting path P_i is found. We augment the matching M^* along this path. Note that, the edges that newly enter the offline matching M^* satisfy (2.3). To ensure that any such newly entered edge (s, r) satisfies (2.2) and therefore the matching remains t-feasible, we will reduce the dual weight of the request by the quantity $(t - 1)d(s, r)$. Both phases can be implemented in $O(n^2)$ time. Details of the running time and correctness proof for this algorithm can be found in the paper by Raghvendra [12]. In addition, it is also shown that the algorithm maintains the following three invariants where S_F represents the set of free servers:

(I1) M^* and the dual weights $y(\cdot)$ form a t-feasible matching,

(I2) For every server $s \in S$, $y(s) \leq 0$, and, if $s \in S_F$, $y(s) = 0$ and for every request $r \in R$, $y(r) \geq 0$ and if r has not yet arrived, $y(r) = 0$.

(I3) When a request r_i is processed and an augmenting path P_i is found, the dual weight $y(r_i)$ is equal to the t-net-cost $\phi_t(P_i)$.

Throughout the rest of this thesis, we will use the following notations. We will set t to be an approximately chosen large constant. We will index the requests in the order of their arrival, i.e., let r_i be the ith request to arrive. Let R_i be the set of first i request locations. Let $\sigma(R) = \langle r_1, ..., r_i, ..., r_n \rangle$ be this sequence of all the n requests where r_i arrived before r_j if $i < j$. For any request r, let $h(r)$ be the index of this request in $\sigma(R)$. For any subset of requests $R' \subseteq R$, we can define the sequence $\sigma(R') = \langle r'_1, \ldots, r'_{|R'|} \rangle$ in a similar fashion where r'_i appears before r'_j in $\sigma(R')$ if and only if $h(r'_i) < h(r'_j)$. While processing a request r_i from $\sigma(R)$, our algorithm will compute a minimum t-net-cost augmenting path P_i. Let the t-net-cost of P_i be $\phi_i (= \phi_t(P_i))$. Let $\sigma(P) = \langle P_1, ..., P_n \rangle$ be this sequence of augmenting paths generated by the algorithm. We denote the free server at the other end of the P_i by s_i. Let M_i^* be the offline matching after the ith request has been processed; i.e., the matching obtained after augmenting the matching M_{i-1}^* along P_i. Note that M_0^* is an empty matching and $M_n^* = M^*$ is the final matching after all the n requests have been processed. The matching M_i will denote the online matching produced by the algorithm for the first i requests. M_i consists of edges $\bigcup_{j=1}^{i} (s_j, r_j)$. Let S_i^F be the free servers with respect to matchings M and M^* after processing i requests. For any path P, recollect that $w(P) = \sum_{(s,r) \in P} d(s, r)$ is its *cost*.

The following properties of the algorithm will be useful in the analysis:

(P1) The cost of any offline matching $w(M_i^*)$ is at most $tw(M_{\text{OPT}})$, and

(P2) For any augmenting path P_i computed by our algorithm, $\phi_t(P_i) \leq tw(M_{\text{OPT}})$.

The following Lemma from [12] establishes that we can bound the cost of the online matching with the sum of the t-net-cost of all augmenting paths generated by the algorithm.

Lemma 1 *Let $t \geq 1$. Let $P_1, \ldots, P_n$ be the augmenting paths computed by our algorithm in that order. Then, the t-net-cost of these paths relate to the cost of the online matching in the following way:*

$$\sum_{i=1}^{n} \phi_t(P_i) \geq ((t-1)/2)w(M) + ((t+1)/2)w(M_{\text{OPT}}).$$

2.1 Properties of the Algorithm

The following properties of the **RM**-Algorithm will be useful in the analysis:

(P1) The offline matching M^* maintained by the algorithm is at most $tw(M_{\text{OPT}})$,

(P2) For any augmenting path P_i computed by our algorithm, the t-net-cost of the augmenting path P_i is at most $tw(M_{\text{OPT}})$.

We include the proof of these properties below:

Lemma 2 *For any augmenting path P_i computed by our algorithm, the t-net-cost of the augmenting path P_i is at most $tw(M_{\text{OPT}})$.*

Proof: Let us assume that M_{OPT} is an optimal matching of the sets S and R. For the ith request r_i, let M^*_{i-1} be the offline matching maintained by the algorithm just before request r is processed. Consider the graph $G(S \cup R, M_{\text{OPT}} \oplus M^*_{i-1})$. Since M_{OPT} is a perfect matching, graph G contains a set of $n - i + 1$ vertex disjoint augmenting paths, each with one of the $n - i + 1$ remaining requests as an end vertex. Let P' be the augmenting path in G that has r_i as one of its end vertex and let its t-net-cost be ϕ'. By the definition of t-net-cost of the augmenting path,

$$\phi' = t \sum_{(s,r) \in M_{\text{OPT}} \cap P'} d(s,r) - \sum_{(s,r) \in P' \cap M^*_{i-1}} d(s,r) \leq t \sum_{(s,r) \in M_{\text{OPT}} \cap P'} d(s,r) \leq tw(M_{\text{OPT}}).$$

Note that the augmenting path P_i computed by our algorithm while processing request r_i is a minimum t-net-cost augmenting path with a t-net-cost of ϕ_i. Therefore,

$$\phi_i \leq \phi' \leq tw(M_{\text{OPT}})$$

as desired. □

Lemma 3 *The cost of offline matching M^* maintained by the algorithm is at most $tw(M_{\text{OPT}})$.*

Proof: Let M^* to be the offline matching computed by the algorithm after all the requests in set R have been processed. Let M_{OPT} be the optimal matching for the set S and R. Note that both the matching M^* and M_{OPT} are perfect matchings. Consider the graph $G(S \cup R, M_{\text{OPT}} \oplus M^*)$. This graph will consist of alternating cycles. For any such alternating cycle C, the t-net-cost is given by

$$\phi(C) = \left(t \sum_{(r,s) \in C \setminus M^*} d(r,s) \right) - \sum_{(r,s) \in C \cap M^*} d(r,s) \geq 0$$

$$\begin{aligned} \sum_{(r,s) \in C \cap M^*} d(r,s) &\leq \left(t \sum_{(r,s) \in C \cap M_{\text{OPT}}} d(r,s) \right) \\ w(M^*) &\leq t(w(M_{\text{OPT}})), \end{aligned}$$

as desired. □

Chapter 3

Input Sensitive Analysis

In this chapter, we will analyze the **RM**-Algorithm for any given input instance, i.e., metric space and server locations. For the input sensitive analysis, we will consider the value of t as a constant with respect to number of servers and requests. In Chapter 4, we also provide the lower bound constructions which show the affect of value of t on the performance of the algorithm.

For the analysis, we say that any request or server that is unmatched at the current time step is considered to be a *free vertex*. We also consider that any path is a sequence of nodes where any two adjacent nodes are connected by an edge, i.e, if P is a path denoted by $P = \langle v_1, v_2, ..., v_p \rangle$, then every pair of vertices v_i and v_{i+1} is connected by an edge for $1 \leq i \leq p-1$.

Next, we begin by describing a variant of the Vitali's Covering Lemma that we use for the analysis of the **RM**-Algorithm.

3.1 A variant of Vitali's Covering Lemma

In order to bound the competitive ratio of our algorithm, we introduce a variant of *Vitali's Covering Lemma* [13]. Given a ball $\mathcal{B}$ with a center c and radius $\mathbf{r}$, let $3\mathcal{B}$ denote the 3-*expansion* of ball $\mathcal{B}$ which is a ball with center c and radius $3\mathbf{r}$. Given a set of balls B, Vitali's covering lemma states

that it is possible to select a subset $B' \subseteq B$ of disjoint balls such that the union of their 3-expansion covers every ball in B.

In our setting, we consider a set $B = \{\mathcal{B}_1, \ldots, \mathcal{B}_n\}$ of weighted balls, all with the same radius $\mathbf{r}$. The weight of any ball $\mathcal{B}_i \in B$ is given by some $\beta_i > 0$. We present a greedy procedure to select a set $B' = \{\mathcal{B}_{s_1}, \mathcal{B}_{s_2}, \ldots, \mathcal{B}_{s_m}\}$ where $B' \subseteq B$ and has the following properties (Lemma 4):

(V1) For any i, j, $\mathcal{B}_{s_i} \cap \mathcal{B}_{s_j} = \emptyset$,

(V2) $(\mathcal{B}_1 \cup \mathcal{B}_2 \cup ... \cup \mathcal{B}_n) \subseteq (3\mathcal{B}_{s_1} \cup ... \cup 3\mathcal{B}_{s_m})$, and,

(V3) For every ball $\mathcal{B}_i \in B$ with weight β_i, there exists a ball $\mathcal{B}_{s_j} \in B'$ with a weight β_{s_j} such that $\beta_{s_j} \geq \beta_i$ and $\mathcal{B}_i \cap \mathcal{B}_{s_j} \neq \emptyset$.

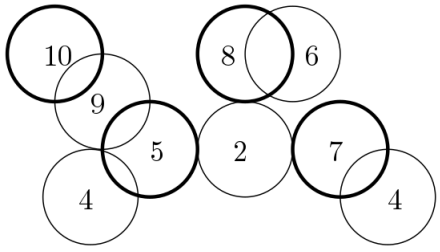

Figure 3.1. An example of the variant of Vitali's covering lemma. Every ball has the same radius; highlighted balls are selected by the procedure.

Selection Procedure. Initialize $\hat{B}$ to B and B' to $\emptyset$. We select the subset of balls $B' = \{\mathcal{B}_{s_1}, \ldots, \mathcal{B}_{s_m}\}$ in an iterative fashion as follows: At the start of iteration i, let $B' = \{\mathcal{B}_{s_1}, \ldots, \mathcal{B}_{s_{i-1}}\}$ be the set of disjoint balls computed by the algorithm. Let $\mathcal{B}_{s_i}$ be the ball with the largest weight in $\hat{B}$. We add $\mathcal{B}_{s_i}$ to B'. Let $B_i \subseteq \hat{B}$ be the largest set of balls such that every ball in this set has a non-empty intersection with $\mathcal{B}_{s_i}$. We set $\hat{B} \leftarrow \hat{B} \setminus B_i$ and repeat this procedure until $\hat{B} = \emptyset$. We call the set of balls B_i as the *intersecting set* of $\mathcal{B}_{s_i}$and denote it by $\mathcal{IS}_{s_i}$. Property (V1) follows from this selection procedure in a straightforward way. The next lemma will establish property (V2) and (V3) for the set B' computed by this procedure.

Lemma 4 *Given a set of weighted balls B, the set B' computed by the selection procedure described above will satisfy (V2) and (V3).*

Proof: For every ball $\mathcal{B}_i \in B$, we will show that there is a ball $\mathcal{B}_{s_j} \in B'$ such that $\mathcal{B}_i \subset 3\mathcal{B}_{s_j}$ and (V2) immediately follows. Recollect that the radius of every ball $\mathcal{B} \in B$ is $\mathbf{r}$. Also note that $\hat{B} = B$ at the start of the algorithm and $\hat{B} = \emptyset$ at the end. Therefore, there is a ball $\mathcal{B}_{s_j}$ selected by the algorithm such that $\mathcal{B}_i \in \mathcal{IS}_{s_j}$. Since every ball in the intersecting set including $\mathcal{B}_i$ intersects $\mathcal{B}_{s_j}$, there is at least one common point to both $\mathcal{B}_i$ and $\mathcal{B}_{s_j}$. As $\mathcal{B}_i$ and $\mathcal{B}_{s_j}$ have the same radius $\mathbf{r}$, the farthest point in B_i from the common point is at most a distance $2\mathbf{r}$. Similarly, the common point is at most at a distance $\mathbf{r}$ from the center of the ball $\mathcal{B}_{s_j}$. From the triangle inequality, the farthest point in $\mathcal{B}_i$ is at most $3\mathbf{r}$ from the centre of $\mathcal{B}_{s_j}$ and we have $\mathcal{B}_i \subset 3\mathcal{B}_{s_j}$.

To prove (V3), let $\mathcal{B}_{s_j}$ be the ball selected in some iteration j such that the ball $\mathcal{B}_i \in \mathcal{IS}_{s_j}$. At the start of this iteration, $\mathcal{B}_i$ was in the set $\hat{\mathcal{B}}$. Since the ball $\mathcal{B}_{s_j}$ was chosen in iteration j, it had the largest weight among all balls of $\hat{B}$ including $\mathcal{B}_i$. Therefore, $\beta_i \leq \beta_{s_j}$. □

3.2 Analysis of the algorithm

For a point set W, let $\textsc{Tsp}(W)$ denote the cost of the minimum length tour that starts and ends at the same point $s \in W$ and visits every other point in W exactly once; here the cost of a tour is the sum of the costs of the individual edges that participate in the tour. Let the distance between the farthest pair of points in W be the diameter of W, and denote it by $\textsc{Diam}(W)$, i.e., $\textsc{Diam}(W) = \max_{p,q \in W} d(p, q)$. Let the stretch of point set W, denoted by $\mu_{\mathbb{M}}(W)$, be the ratio of $\textsc{Tsp}(W)$ and $\textsc{Diam}(W)$ for the point set W.

$$\mu_{\mathbb{M}}(W) = \frac{\textsc{Tsp}(W)}{\textsc{Diam}(W)}.$$

For a given metric space $\mathbb{M}$ and any positive integer n, we define the *stretch* of the metric space $\mathbb{M}$, denoted by $\mu_{\mathbb{M}}(n)$, to be the largest stretch of any n-point set W from this metric space.

$$\mu_{\mathbb{M}}(n) = \max_{\substack{W \subset \mathbb{M} \\ |W|=n}} \mu_{\mathbb{M}}(W).$$

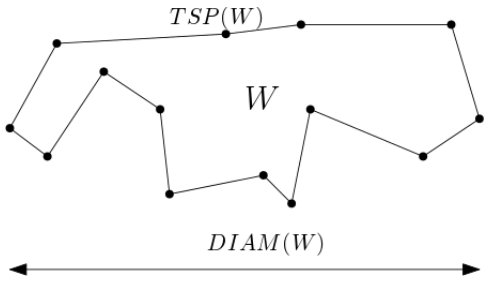

Figure 3.2. The figure shows TSP(W) and DIAM(W) of the point set

In this section, we will show that, for some set $S' \subseteq S$, the **RM**-Algorithm achieves a competitive ratio of $O((\log^2 n)\mu_{\mathbb{M}}(S'))$.

3.2.1 Overview of the Input Sensitive Analysis

For each request r_i, we place a ball which we refer to as the inner-ball $\mathcal{IB}_i$, centered at this request and having a radius that is proportional to the t-net-cost $\phi_t(P_i)$. We also construct an outer-ball $\mathcal{OB}_i$ which is also a ball centered at the request and that has a radius larger than the inner ball. Any outer ball is carefully constructed to contain a high density of edges from the optimal matching M_{OPT}. To obtain an upper bound on the online cost, we have to estimate the sum of the radii of the inner balls (Lemma 1). To obtain a lower bound on the optimal matching M_{OPT}, we can add the edges of M_{OPT} belonging to any set of disjoint outer balls such as the one produced by Vitali's Lemma. Therefore, the problem of computing the competitive ratio reduces to relating the inner and outer balls.

We partition the requests R into $O(\log n)$ "inner groups" such that any two requests in the same group have a similar inner ball radius. We analyze each such group.

For an inner group $\mathcal{IG} \subseteq R$, let every request have an inner ball radius that is approximately $\mathbf{r}$. We present an intuition for the special case where every pair of requests of the same group $\mathcal{IG}$ have a distance of at least $\mathbf{r}/c$, for some constant c. We apply Vitali's covering lemma on the set of outer balls of the requests in $\mathcal{IG}$ and identify a set of disjoint outer balls and an intersecting set for each outer ball of this set. We switch the outer balls in every intersecting set with the corresponding inner

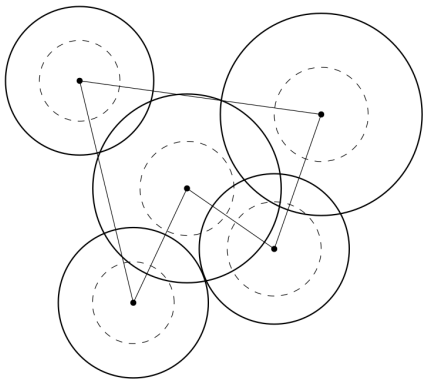

Figure 3.3. For every request shown, there is an inner ball and an outer ball. Note that the sum of the radii of the inner balls is bounded by the cost of the traveling salesman tour.

balls. Due to the large separation between their centers, we can bound the sum of the inner ball radius in the intersecting set by the cost of a traveling salesman tour on its centers. We establish a lower bound on the optimal matching by relating the radius of the outer ball and the diameter of the intersecting set (by Vitali's Lemma) and relating the radius of the outer ball with the optimal matching.

However, the assumption of large separation between requests is not true. We overcome this by carefully clustering the requests such that their centers have a large separation. Such clusters create challenges in defining outer balls. We also partition requests based on their outer ball radius into $O(\log n)$ "outer groups", two requests with roughly equal outer-ball radius will belong to the same outer group. We then apply the variant of Vitali's covering lemma for all requests that have the same inner group and outer group. Since there are $O(\log^2 n)$ combinations of inner groups and outer groups, we get an additional factor of $\log^2 n$ in our analysis.

3.2.2 Inner groups, Clusters, and Outer groups

Fix an optimal matching M_{OPT} of the requests R and servers S. For any point $v \in S \cup R$, we define $M_{\text{OPT}}(v)$ to be the edge of M_{OPT} that is incident on the vertex v. For any subset $S' \subseteq S$ (resp, $R' \subseteq R$), we define $M_{\text{OPT}}(S')$ (resp, $M_{\text{OPT}}(R')$) to be the set of edges of M_{OPT} incident on vertices

of S' (resp, R'). Also, we denote the sequence of any set by $\sigma(.)$. For our analysis, we partition all the requests into $O(\log n)$ *inner groups* and into $O(\log n)$ *outer groups*. We partition requests into an inner group as follows:

- For any request r_j, if $\phi_j < \frac{w(M_{\text{OPT}})}{n}$ or if $\phi_j \leq 16tw(M_{\text{OPT}}(r_j))$, then the request is assigned to inner group $\mathcal{IG}_0$,
- For the requests that are not assigned to inner group $\mathcal{IG}_0$, if $\phi_j \geq \frac{w(M_{\text{OPT}})}{n}$ and $\frac{2^{i-1}w(M_{\text{OPT}})}{n} \leq \phi_j < \frac{2^i w(M_{\text{OPT}})}{n}$, then the request is assigned to inner group $\mathcal{IG}_i$.

In this way, we partition all the requests of R into $m+1$ non-empty groups $\mathcal{IG}_0, \mathcal{IG}_1, \ldots, \mathcal{IG}_m$. In Lemma 5, we show that number of groups so formed is $O(\log n)$. In our analysis of the algorithm, we deal with the inner group 0 separately. In the following, we describe the construction of inner balls for every inner group. However, we construct outer balls only for groups $\mathcal{IG}_1 \ldots \mathcal{IG}_m$.

For $i \geq 1$, let $\mathcal{IG}_i \subseteq R$ be the set of requests that belong to the inner group i with $|\mathcal{IG}_i| = n_i$. Next, we will partition the requests of $\mathcal{IG}_i$ into clusters such that the separation between their centers is proportional to their t-net-costs. For each *cluster* $C \subseteq \mathcal{IG}_i$, we will designate one request as the *representative* of this cluster and denote it by $rc(C)$. We will construct these clusters by processing the requests of $\sigma(\mathcal{IG}_i) = \langle r_1^i, ..., r_{n_i}^i \rangle$ in the order in which they appear in this sequence: Suppose we have already partitioned requests $\langle r_1^i, ..., r_l^i \rangle$ into clusters $\langle C_1^i, ..., C_p^i \rangle$. For the next request r_{l+1}^i, let $k = \arg\min_{j=1,...,p} d(rc(C_j^i), r_{l+1}^i)$ i.e., k is the index of the cluster with the representative request that is closest to r_{l+1}^i and $r' = rc(C_k^i)$.

- If $d(r', r_{l+1}^i) < \frac{2^{i-2}w(M_{\text{OPT}})}{tn}$, then we assign the request r_{l+1}^i to cluster C_k^i,
- Otherwise, we create a new cluster $C_{p+1}^i = \{r_{l+1}^i\}$ to C^i and set its representative $rc(C_{p+1}^i)$ to be r_{l+1}^i.

Let k_i be the number of clusters formed for all the requests in $\sigma(R_i)$. Let n_k^i be the number of requests in the cluster C_k^i i.e., $|C_k^i| = n_k^i$. For simplicity of notation, we will denote the representative

request $rc(C_k^i)$ by $rc(i,k)$. For any request $r \in C_k^i$, we denote it by $r(i,k,j)$ if r is the jth request in the sequence $\sigma(C_k^i)$. Let $rc(i,k)$ appear at time step h i.e., $h = h(rc(i,k))$. We denote $M(i,k)$ to be the offline matching maintained by the algorithm right before the request $rc(i,k)$ is processed i.e., $M(i,k) = M^*_{h-1}$. From the construction of the clusters and the observation that $rc(i,k)$ is the first request in the sequence $\sigma(C_k^i)$, we observe (C1) and (C2):

(C1) For any cluster C_k^i, the distance of any request $r \in C_k^i$ to its representative is $d(r, rc(C_k^i)) < \frac{2^{i-2}w(M_{\text{OPT}})}{tn}$, and

(C2) Every request in C_k^i is free with respect to the matching $M(i,k)$.

Recollect that we need to construct outer balls that contain a high density of edges from the optimal matching M_{OPT}. We do so for all requests in inner groups $\mathcal{IG}_1, \ldots, \mathcal{IG}_m$. To assist in defining outer balls, for every request $r_g \in R \setminus \mathcal{IG}_0$, we associate an *optimal path* Q_g which is an alternating path containing edges from the optimal matching. Fix any request $r_g \in R$. Let r_g belong to cluster k of group i, i.e., C_k^i . We define the optimal path Q_g for r_g as follows: Consider the graph $\tilde{G} = \tilde{G}(S \cup R, M_{\text{OPT}} \oplus M(i,k))$. Since every request of C_k^i is free in the matching $M(i,k)$(from (C2)) and since M_{OPT} is a perfect matching, $\tilde{G}$ will have a vertex-disjoint augmenting path for request r_g. We set this path as the optimal path Q_g of request r_g. For request r_g, we pick δ_g to be the largest distance between the representative of cluster C_k^i, $rc(i,k)$ and any vertex on the optimal path Q_g, i.e.,

$$\delta_g = \max_{v \in Q_g} d(rc(i,k), v).$$

Next, we partition the requests of $R \setminus \mathcal{IG}_0$ into $O(\log n)$ outer groups. For any request $r_g \in R \setminus \mathcal{IG}_0$, we assign it to an *outer group* i represented by $\mathcal{OG}_i$ if:

$$\frac{2^{i-1}w(M_{\text{OPT}})}{nt} \leq \delta_g < \frac{2^i w(M_{\text{OPT}})}{nt} \qquad \forall \quad 1 \leq i \leq m'. \tag{3.1}$$

Let $R(i,j)$ denote the requests that belong to inner group i and to outer group j. For any cluster C_k^i, let $R(i,j,k)$ denote the requests of $R(i,j)$ that belong to the cluster C_k^i and let $\beta(i,j,k)=$

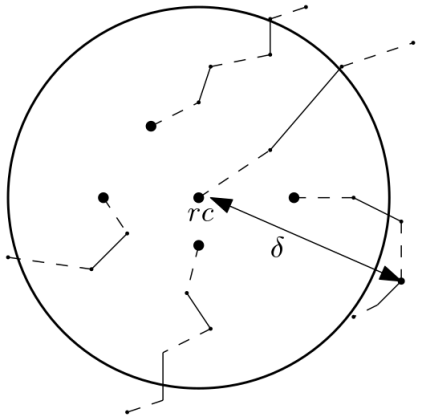

Figure 3.4. The optimal paths for the set of requests in any cluster C with representative request rc.

$|R(i,j,k)|$. We define an *inner ball* and an *outer ball* for each $R(i,j,k)$. An inner ball $\mathcal{IB}(i,j,k)$ for the set $R(i,j,k)$ is centered at the representative request of the cluster C_k^i, i.e., $rc(i,k)$ and has a radius $\frac{2^{i-1}w(M_{\text{OPT}})}{tn}$. Note that the inner radius of any $R(i,j,k)$ depends only the inner group i. Therefore, we denote the radius of the inner ball as the *inner radius* and represented it by $\mathbf{r}_{\mathcal{IB}}(i)$. We also define inner ball for each request in inner group $\mathcal{IG}_0$ as follows: For each request $r \in \mathcal{IG}_0$, we define an inner ball centered at r and with a radius $\frac{\phi_t(r)}{t}$. For $R(i,j,k)$, we define its outer ball, $\mathcal{OB}(i,j,k)$ to be a ball centered at $rc(i,k)$ with a radius $\frac{2^j w(M_{\text{OPT}})}{nt}$. Note that the radius of the outer ball, which we refer to as the *outer radius*, depends only on the outer group and therefore, we represent its radius by $\mathbf{r}_{\mathcal{OB}}(j)$. Note that for any request $r_g \in R(i,j,k)$, by its construction, the optimal path Q_g is contained inside its outer ball $\mathcal{OB}(i,j,k)$.

We say that an edge uv *intersects* a ball $\mathcal{B}$ if both its end points are contained inside $\mathcal{B}$. We define the intersection of the outer ball $\mathcal{OB}$ with any matching M', denoted by $\mathcal{OB} \cap M'$, as those edges of the matching M' that intersect $\mathcal{OB}$.

Next, we will show some properties of the inner and outer groups. The following lemma will establish that there are $O(\log n)$ inner groups and $O(\log n)$ outer groups.

Lemma 5 *The number of inner groups is* $O(\log n)$.

Proof: From Lemma 2, we know that the t-net-cost of every augmenting path computed by the

algorithm is at most $tw(M_{\text{OPT}})$. By construction, $\mathcal{IG}_m$ (where m is the maximum index for inner groups) is non-empty and therefore there will be a request $r_j \in \mathcal{IG}_m$ with a minimum t-net-cost augmenting path of value ϕ_j such that,

$$\frac{2^{m-1}w(M_{\text{OPT}})}{n} \leq \phi_j \leq tw(M_{\text{OPT}}),$$

implying that

$$m \leq \log(nt)$$

and the number of inner groups is $O(\log n)$. □

Lemma 6 *The number of outer groups is* $O(\log n)$.

Proof: To bound the number of outer groups, consider any request $r_g \in R(i,j,k)$. We know that Q_g is an augmenting path that contains edges from $M(i,k)$ and M_{OPT}. From (P1), we know that the cost of the offline matching is at most t times the cost of the optimal matching i.e., $w(M(i,k)) \leq tw(M_{\text{OPT}})$ and therefore $w(Q_g) \leq (t+1)w(M_{\text{OPT}})$.

Since r_g is in inner group i, property (C1) implies that the distance of any request r_g to its representative request $rc(i,k)$ can be written as

$$d(r_g, rc(i,k)) < \frac{2^{i-2}w(M_{\text{OPT}})}{tn} \leq \frac{\phi_g}{2} \leq \frac{tw(M_{\text{OPT}})}{2}.$$

The last inequality follows from (P2). Let δ_g be the distance from the representative request $rc(i,k)$ to the farthest vertex of the path Q_g. By the triangle inequality,

$$\delta_g \leq d(rc(i,k), r_g) + w(Q_g) \leq (t/2)w(M_{\text{OPT}}) + (t+1)w(M_{\text{OPT}}).$$

Using this inequality in (3.1), we obtain

$$\frac{2^{m'}w(M_{\text{OPT}})}{nt} \leq O(w(M_{\text{OPT}})),$$

or $m' \leq O(\log(nt))$ as desired. □

The following lemma shows that the total t-net-costs of the requests in inner group $\mathcal{IG}_0$ is at most $O(w(M_{\text{OPT}}))$.

Lemma 7 *The sum of the t-net cost of all requests $r \in \mathcal{IG}_0$ satisfies*

$$\sum_{r \in \mathcal{IG}_0} \phi_t(r) \leq (16t+1)w(M_{\text{OPT}}).$$

Proof: According to the definition of inner groups, we know that for any $r \in \mathcal{IG}_0$, either $\phi < \frac{w(M_{\text{OPT}})}{n}$ or $\phi \leq 16tw(M_{\text{OPT}}(r_j))$. We bound the sum of the t-net cost of the requests following each condition separately. For every request r_j in $R' \subseteq \mathcal{IG}_0$, let $\phi_j < \frac{w(M_{\text{OPT}})}{n}$. Then, we know that $|R'| \leq n$ and, hence

$$\sum_{r_j \in R'} \phi_i < w(M_{\text{OPT}}).$$

Next, for any request $r_{j'}$ in $R'' \subseteq \mathcal{IG}_0$, let $\phi_j \leq 16tw(M_{\text{OPT}}(r_j))$. Then

$$\begin{aligned} \sum_{r_{j'} \in R''} \phi_j &\leq \sum_{r_{j'} \in R''} 16tw(M_{\text{OPT}}(r_j)) \\ &\leq 16tw(M_{\text{OPT}}) \end{aligned}$$

Hence,

$$\sum_{r \in \mathcal{IG}_0} \phi \leq (16t+1)w(M_{\text{OPT}}),$$

as desired. □

3.2.3 Properties of inner and outer ball

In this section, we will present several useful properties of inner and outer balls that will help us in the analysis. The following lemma relates the inner radius to both the online matching and the traveling salesman tour.

Lemma 8 *Let the request set $R \setminus \mathcal{IG}_0$ be partitioned into m inner groups and m' outer groups. Let $R(i,j)$ be the requests that have an inner group i and an outer group j. Let the radius of the inner ball $\mathcal{IB}(i,j,k)$ for the request be denoted by $\mathbf{r}_{\mathcal{IB}}(i)$. Suppose M is the online matching produced by our algorithm. Then, the following properties hold:*

(a) $\mathcal{IB}(i,j,k) \cap S_F = \emptyset$*, where* S_F *is the set of free servers with respect to the matching* $M(i,k)$*,*

(b) $\sum_{r \in \mathcal{IG}_0} \frac{\phi_t(r)}{2t} + \sum_{k=0}^{k_i} \sum_{j=1}^{m'} \sum_{i=1}^{m} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i) \geq \frac{t-1}{4t} w(M)$*,*

(c) For $i \geq 1$*, let* $\tilde{R}$ *be any subset of representative requests for the clusters in inner group* $\mathcal{IG}_i$ *and let* $\tilde{S}$ *be the servers that match to the requests of* $\tilde{R}$ *in* M_{OPT} *. Then* $|\tilde{R}|\frac{\mathbf{r}_{\mathcal{IB}}(i)}{4} \leq \text{TSP}(\tilde{S})$*.*

Proof:

(a) Let r' be the representative request $rc(i,k)$ of cluster C_k^i. For the sake of contradiction, let us assume that the inner ball contains a free server $s \in S_F \cap \mathcal{IB}(i,j,k)$ i.e, distance between representative request r' and s is $d(s,r') < \frac{2^{i-1}w(M_{\text{OPT}})}{nt} \leq \frac{\phi_t(r')}{t} \leq \frac{y(r')}{t}$. Here $y(r')$ is the dual weight of r' when the algorithm processes r' and computes an augmenting path (from (I3)). Therefore,

$$y(r') > td(s,r').$$

By the feasibility condition (2.1), we know that $y(r')+y(s) \leq td(s,r')$ implying that $y(s) < 0$ contradicting invariant (I2).

(b) By the definition of inner group $\mathcal{IG}_i$ for $i \geq 1$, we know that $\phi_g < \frac{2^i w(M_{\text{OPT}})}{n}$, where ϕ_g is the t-net-cost of the augmenting path for the request r_g computed by the algorithm. Dividing both the sides of the above equation by $2t$,

$$\frac{\phi_g}{2t} < \frac{2^{i-1}w(M_{\text{OPT}})}{tn}$$

$$\frac{\phi_g}{2t} < \mathbf{r}_{\mathcal{IB}}(i)$$

Since every request that is not in inner group $\mathcal{IG}_0$ belongs to a unique inner group $i \geq 1$, outer group j and cluster k, and every request in $R(i,j,k)$ has the same inner radius, we get

$$\sum_{r_g \in R \setminus \mathcal{IG}_0} \frac{\phi_g}{2t} < \sum_{k=0}^{k_i} \sum_{j=1}^{m'} \sum_{i=1}^{m} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i) \tag{3.2}$$

Also, from Lemma 1, we know that

$$\frac{t-1}{2}w(M) + \frac{t+1}{2}w(M^*) \leq \sum_{r_g \in R} \phi_g \tag{3.3}$$

$$\frac{t-1}{2}w(M) \leq \sum_{r_g \in R} \phi_g \tag{3.4}$$

$$\leq \sum_{r_g \in \mathcal{IG}_0} \phi_g + \sum_{r_g \in R \setminus \mathcal{IG}_0} \phi_g \tag{3.5}$$

Hence, from equations (3.2) and (3.5),

$$\sum_{r \in \mathcal{IG}_0} \frac{\phi_t(r)}{2t} + \sum_{k=0}^{k_i} \sum_{j=0}^{m'} \sum_{i=0}^{m} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i) \geq \frac{t-1}{4t}w(M)$$

as desired.

(c) Let the smallest cost tour that visits every representative request of $\tilde{R}$ be $T = \langle \tilde{r}_1, \tilde{r}_2, \ldots, \tilde{r}_{|\tilde{R}|}, \tilde{r}_1 \rangle$. By the construction of our clusters, the distance between any two representative requests is at least $d(\tilde{r}_i, \tilde{r}_j) \geq \frac{2^{i-2}w(M_{\text{OPT}})}{tn}$, since otherwise we would place the two requests in the same cluster. Hence,

$$\text{TSP}(\tilde{R}) = d(\tilde{r}_1, \tilde{r}_2) + ... + d(\tilde{r}_{|\tilde{R}|}, \tilde{r}_1), \tag{3.6}$$

$$\geq |\tilde{R}|\frac{2^{i-2}w(M_{\text{OPT}})}{tn} \geq |\tilde{R}|\frac{\mathbf{r}_{\mathcal{IB}}(i)}{2}. \tag{3.7}$$

From the definition of inner group i, for any $r_g \in \mathcal{IG}_i$ we know that,

$$\phi_g < \frac{2^i w(M_{\text{OPT}})}{n}, \text{and} \quad \phi_g > 16tw(M_{\text{OPT}}(r_g))$$

$$16tw(M_{\text{OPT}}(r_g)) < \frac{2^i w(M_{\text{OPT}})}{n}$$

$$8w(M_{\text{OPT}}(r_g)) < \frac{2^{i-1}w(M_{\text{OPT}})}{tn} < \mathbf{r}_{\mathcal{IB}}(i)$$

$$\sum_{r_g \in \tilde{R}} 8w(M_{\text{OPT}}(r_g)) < \sum_{r_g \in \tilde{R}} \mathbf{r}_{\mathcal{IB}}(i)$$

$$8w(M_{\text{OPT}}(\tilde{R})) < |\tilde{R}|\mathbf{r}_{\mathcal{IB}}(i)$$

Also, we can say that

$$\text{TSP}(\tilde{R}) \leq \text{TSP}(\tilde{S}) + 2w(M_{\text{OPT}}(\tilde{R})) \tag{3.8}$$

From equations (3.7) and (3.8), we have

$$\begin{aligned} |\tilde{R}|\frac{\mathbf{r}_{\mathcal{JB}}(i)}{2} &\leq \textsc{Tsp}(\tilde{S}) + 2w(M_{\textsc{Opt}}(\tilde{R})) \leq \textsc{Tsp}(\tilde{S}) + |\tilde{R}|\frac{\mathbf{r}_{\mathcal{JB}}(i)}{4} \\ |\tilde{R}|\frac{\mathbf{r}_{\mathcal{JB}}(i)}{4} &\leq \textsc{Tsp}(\tilde{S}), \end{aligned}$$

as desired.

□

Lemma 9 *For $i, j \geq 1$, let $R(i,j,k)$ be the subset of requests that have inner group i and outer group j and belong to the cluster C_k^i. Let $\mathbb{Q}$ be the set of optimal paths of requests in $R(i,j,k)$. Then:*

(a) $\mathbb{Q}$ is a set of vertex-disjoint augmenting paths,

(b) For $Q \in \mathbb{Q}$, $\ell(Q) \geq \frac{2^{j-2}w(M_{\textsc{Opt}})}{tn}$, and

(c) For any optimal path $Q \in \mathbb{Q}$, $\frac{w(M_{\textsc{Opt}} \cap Q)}{w(Q)} \geq \frac{1}{t+1}$.

Proof: For (a), let $R(i,k)$ be the requests of inner group i and cluster k. By our construction, $M(i,k)$ is the matching before we processes any request in $R(i,k)$, and hence every request of $R(i,j,k) \subset R(i,k)$ is free with respect to the matching $M(i,k)$. Therefore, there is a vertex-disjoint augmenting path for each request of $R(i,j,k)$ in $\tilde{G} = \tilde{G}(S \cup R, M_{\textsc{Opt}} \oplus M(i,k))$. This, by construction, is precisely the set of vertex-disjoint augmenting path in $\mathbb{Q}$.

For (b), let the optimal path for request $r \in C_k^i$ be Q and let $rc = rc(i,k)$ be the representative request for the cluster C_k^i. Let δ be the distance from rc to the farthest vertex on Q. By the triangle inequality, we know that

$$\delta \leq w(Q) + d(rc, r).$$

Also by the definition of outer radius, $\frac{2^{j-1}w(M_{\textsc{Opt}})}{nt} \leq \delta$ and $d(rc,r) \leq \frac{2^{i-2}w(M_{\textsc{Opt}})}{nt}$.

We can write the above equation as,

$$\frac{2^{j-1}w(M_{\text{OPT}})}{nt} - \frac{2^{i-2}w(M_{\text{OPT}})}{nt} \leq w(Q).$$

In Lemma 10, we show that the inner group i of a request is strictly smaller than its outer group j. Therefore,

$$w(Q) \geq \frac{2^{j-1}w(M_{\text{OPT}})}{nt} - \frac{2^{j-2}w(M_{\text{OPT}})}{nt} \geq \frac{2^{j-2}w(M_{\text{OPT}})}{nt},$$

as desired.

For (c), let Q be any optimal path for the request $r \in C_k^i$. Note that Q is an augmenting path with respect to the matching $M(i,k)$ and contains edges of M_{OPT}. Let s be the free server at the other end of Q. For the sake of contradiction, let us assume

$$\begin{aligned}
\frac{w(Q \cap M_{\text{OPT}})}{w(Q)} &< \frac{1}{t+1} \\
(t+1)w(Q \cap M_{\text{OPT}}) &< w(Q) \\
tw(Q \setminus M(i,k)) &< w(Q) - w(Q \cap M_{\text{OPT}}) = w(Q \cap M(i,k)) \\
t(w(Q \setminus M(i,k) - w(Q \cap M(i,k) &< 0,
\end{aligned}$$

implying that $\phi_t(Q) < 0$. Using dual weights, we can express $\phi_t(Q)$ as

$$\begin{aligned}
\phi_t(Q) &\leq t \sum_{(s',r') \in Q \setminus M(i,k)} d(s',r') - \sum_{(s',r') \in Q \cap M(i,k)} d(s',r') \\
&\leq \sum_{(s',r') \in Q \setminus M(i,k)} (y(s') + y(r')) - \sum_{(s',r') \in Q \cap M(i,k)} (y(s') + y(r'))
\end{aligned}$$

The first inequality follows from the t feasibility conditions 2.1 and 2.2 . Every vertex of Q except s and r cancel each other in the last inequality, and, since $y(r) = 0$, we obtain $y(s) < 0$ contradicting (I2). □

Lemma 10 *For any request r, if r belongs to inner group $i \geq 1$ and outer group $l \geq 1$, then $l \geq i$ and the inner ball radius of r is strictly smaller than its outer ball radius.*

Proof: Let us assume that the request $r_j \in C_k^i$ has the inner ball centered at representative request rc with radius $\mathbf{r}_{\mathcal{IB}}$ and the outer ball centered at representative request rc with radius $\mathbf{r}_{\mathcal{OB}}$. Let δ_j be the distance of farthest vertex on its optimal path Q_j. The optimal path will start at r_j and end at any server $s_j \in S_F$. As per the definition of inner ball, if ϕ_j lies in the given range $\frac{2^{i-1}w(M_{\text{OPT}})}{n} \leq \phi_j < \frac{2^i w(M_{\text{OPT}})}{n}$ and $\mathbf{r}_{\mathcal{IB}}$ is set to $\frac{2^{i-1}w(M_{\text{OPT}})}{nt}$. From Lemma 8(a), we know that s_j lies outside the inner ball and therefore $\delta_j \geq \frac{2^{i-1}w(M_{\text{OPT}})}{tn}$. The outer group of r_j is l if $\frac{2^{l-1}w(M_{\text{OPT}})}{nt} \leq \delta_j < \frac{2^l w(M_{\text{OPT}})}{nt}$ holds, and therefore $l \geq i$. By its definition, the outer radius $\mathbf{r}_{\mathcal{OB}}$ is set to $\frac{2^l w(M_{\text{OPT}})}{nt} > \mathbf{r}_{\mathcal{IB}}$. □

Lemma 11 *Let r be a request in cluster k, outer group j and inner group i. Let $\beta(i,j,k)$ be the number of requests in cluster k and outer group j and inner group i. Let $\mathcal{OB}$ be the outer ball of request r with radius $\mathbf{r}_{\mathcal{OB}}$. Then $w(\mathcal{OB} \cap M_{\text{OPT}}) \geq \frac{4}{(t+1)}\beta(i,j,k)\mathbf{r}_{\mathcal{OB}}$.*

Proof: Let Q be the optimal path for request r in outer group j and inner group i. We know that $\mathbf{r}_{\mathcal{OB}} = \frac{2^j w(M_{\text{OPT}})}{nt}$ for any outer group j. From Lemma 9(b), we can say that $w(Q) \geq \frac{2^{j-2}w(M_{\text{OPT}})}{tn}$, which implies

$$\begin{aligned} w(Q) &\geq 4\mathbf{r}_{\mathcal{OB}} \\ (t+1)w(M_{\text{OPT}} \cap Q) &\geq 4\mathbf{r}_{\mathcal{OB}} \\ w(M_{\text{OPT}} \cap Q) &\geq \frac{4}{t+1}\mathbf{r}_{\mathcal{OB}} \end{aligned}$$

The second to last inequality holds from Lemma 9(c). By construction Q is contained inside $\mathcal{OB}$. Since optimal path of each of the $\beta(i,j,k)$ requests in $R(i,j,k)$ is vertex-disjoint and contained inside $\mathcal{OB}$. Therefore, we can bound the edges of the optimal matching inside $\mathcal{OB}$ by

$$w(M_{\text{OPT}} \cap \mathcal{OB}) \geq \frac{4}{t+1}\beta(i,j,k)\mathbf{r}_{\mathcal{OB}},$$

as desired. □

3.2.4 Analysis

To bound the competitive ratio of the algorithm, we will first provide the following bound for the request set $R(i,j)$:

$$\frac{\sum_{k=1}^{k_i} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i)}{w(M_{\text{OPT}})} \leq O(\mu_{\mathbb{M}}(S_{i,j})), \tag{3.9}$$

where $S_{i,j}$ is some subset of S. We add (3.9) for each inner group and outer group and use Lemma 8 and Lemma 7 to obtain the following bound on the competitive ratio of our algorithm.

$$\begin{aligned}
\frac{w(M)}{w(M_{\text{OPT}})} &\leq \frac{4t}{t-1}\frac{\sum_{r\in\mathcal{IG}_0}\frac{\phi_t(r)}{2t} + \sum_{j=1}^{m'}\sum_{i=1}^{m}\sum_{k=1}^{k_i}\beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i)}{w(M_{\text{OPT}})} \\
&\leq \frac{4t}{t-1}\left(\frac{\frac{16t+1}{2t}w(M_{\text{OPT}}) + \sum_{j=1}^{m'}\sum_{i=1}^{m}\sum_{k=1}^{k_i}\beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i)}{w(M_{\text{OPT}})}\right) \\
&\leq \frac{32t+2}{t-1} + \frac{4t}{t-1}\sum_{i=1}^{m}\sum_{j=1}^{m'} O(\mu_{\mathbb{M}}(S_{i,j})) \leq O(\mu_{\mathbb{M}}(S')\log^2 n),
\end{aligned}$$

where $S' = \arg\max_{i,j}\mu_{\mathbb{M}}(S_{i,j})$.

Therefore, we prove (3.9) to complete our argument.

Lemma 12 *Let $R(i,j)$ be the requests in any inner group i and outer group j for $i \geq 1$. Then, for some $S_{i,j} \subseteq S$,*

$$\frac{\sum_{k=1}^{k_i} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i)}{w(M_{\text{OPT}})} \leq O(\mu_{\mathbb{M}}(S_{i,j})).$$

Proof: Note that all requests of $R(i,j)$ belong to one of the clusters in $\{C_1^i, \ldots, C_{k_i}^i\}$. Also, by our construction, every cluster has a unique inner ball and an unique outer ball. Let B be the set of outer balls of every cluster for which $\beta(i,j,k) \geq 1$. Using $\beta(i,j,k)$ values as the weights, we apply our procedure described in the variant of Vitali's covering Lemma, i.e., Lemma 4 to obtain a set $B'$of disjoint outer balls. By property (V1), every ball in B' is disjoint and from Lemma 11, we know that every such ball B_{s_i} with weight β_{s_i} will intersect edges of the optimal matching with

$w(M_{\text{OPT}} \cap \mathcal{B}_{s_i}) \geq \frac{4}{(t+1)}\beta_{s_i}\mathbf{r}_{\mathcal{OB}}$. Therefore, we can relate the radius of the outer ball to the total cost of the optimal matching

$$w(M_{\text{OPT}}) \geq \sum_{\mathcal{B}_{s_i} \in B'} \frac{4}{(t+1)}\beta_{s_i}\mathbf{r}_{\mathcal{OB}} = \sum_{\mathcal{B}_{s_i} \in B'} \kappa_{s_i}. \tag{3.10}$$

Here $\kappa_{s_i} = \frac{4}{(t+1)}\beta_{s_i}\mathbf{r}_{\mathcal{OB}}$. The procedure for the variant of Vitali's Lemma also partitions B into intersecting sets, one set $\mathcal{IS}_{s_i}$ for every selected ball $\mathcal{B}_{s_i} \in B'$. Consider one such selected ball $\mathcal{B}_{s_i} \in B'$. Consider its intersecting set. Let K_{s_i} be the index of all the clusters whose outer ball participates in $\mathcal{IS}_{s_i}$. Let $\theta_{s_i} = \sum_{k \in K_{s_i}} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i)$. Therefore, we can express

$$\sum_{k=1}^{k_i} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i) = \sum_{\mathcal{B}_{s_i} \in B'} \sum_{k \in K_{s_i}} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i) = \sum_{\mathcal{B}_{s_i} \in B'} \theta_{s_i}.$$

Using the above equation and (3.10), we can rewrite (3.9) as

$$\frac{\sum_{k=1}^{k_i} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i)}{w(M_{\text{OPT}})} \leq \frac{\sum_{\mathcal{B}_{s_i} \in B'} \theta_{s_i}}{\sum_{\mathcal{B}_{s_i} \in B'} \kappa_{s_i}} = \sum_{\mathcal{B}_{s_i} \in B'} \left(\frac{\kappa_{s_i}}{\sum_{\mathcal{B}_{s_j} \in B'} \kappa_{s_j}} \right) \frac{\theta_{s_i}}{\kappa_{s_i}}$$

Therefore, $\frac{\sum_{\mathcal{B}_{s_i} \in B'} \theta_{s_i}}{\sum_{\mathcal{B}_{s_i} \in B'} \kappa_{s_i}}$ is simply a weighted average of $\frac{\theta_{s_i}}{\kappa_{s_i}}$. To complete the proof, we will bound the ratio $\frac{\theta_{s_i}}{\kappa_{s_i}}$ by $\mu_{\mathbb{M}}(S_{i,j})$. Let R^{s_i} be the representative requests of every cluster whose outer ball is in $\mathcal{IS}_{s_i}$. From (V3), we know that $\beta_{s_i} = \max_{k \in K_{s_i}} \beta(i,j,k)$ and from (V2) and triangle inequality, we know that $6\mathbf{r}_{\mathcal{OB}} \geq \textsc{Diam}(R^{s_i})$. Consider the edges of the optimal matching M_{OPT} that are incident on R^{s_i}. We call these edges $M^{s_i}_{\text{OPT}}$. Let S^{s_i} be the set of servers at the other end of the $M^{s_i}_{\text{OPT}}$. For every edge $u, v \in M^{s_i}_{\text{OPT}}$, $d(u,v) \leq \frac{2^i w(M_{\text{OPT}})}{tn}$. As all the edges from R^{s_i} to S^{s_i} in the optimal matching are contained inside the inner ball, we can say that $6\mathbf{r}_{\mathcal{OB}} \geq \textsc{Diam}(S^{s_i})$. Also from Lemma 8(c), we know that $|\mathcal{IS}_{s_i}|\mathbf{r}_{\mathcal{IB}}(i) \leq 2\textsc{Tsp}(S^{s_i})$. Also,

$$\begin{aligned} \frac{\theta_{s_i}}{\kappa_{s_i}} &= \frac{\sum_{k \in K_{s_i}} \beta(i,j,k)\mathbf{r}_{\mathcal{IB}}(i)}{\frac{4}{(t+1)}\beta_{s_i}\mathbf{r}_{\mathcal{OB}}} \leq \frac{\beta_{s_i} \sum_{k \in K_{s_i}} \mathbf{r}_{\mathcal{IB}}(i)}{\frac{4}{(t+1)}\beta_{s_i}\mathbf{r}_{\mathcal{OB}}} \\ &\leq \frac{|\mathcal{IS}_{s_i}|\mathbf{r}_{\mathcal{IB}}(i)}{\frac{4}{(t+1)}\mathbf{r}_{\mathcal{OB}}} \leq \frac{O(1)\textsc{Tsp}(S^{s_i})}{\textsc{Diam}(S^{s_i})} \leq O(\mu_{\mathbb{M}}(S^{s_i})) \end{aligned}$$

We set $S_{i,j} = \arg\max_{S^{s_i}, B_{s_i} \in B'} \mu_{\mathbb{M}}(S^{s_i})$. From this choice, we have $\frac{\theta_{s_i}}{\kappa_{s_i}} \leq O(\mu_{\mathbb{M}}(S_{i,j}))$. $\square$

Chapter 4

Lower Bounds

In this chapter, we prove lower bounds on the competitive ratio of the **RM**-Algorithm. It is known that the optimal performance of the online algorithm for Random Arrival Model and for the Adversarial Model are $2H_n - 1$ and $2n - 1$ respectively and was achieved when $t \to \infty$ [12]. However, in the input sensitive analysis, we need to choose $t = O(1)$ to achieve the best competitive ratio. In Section 4.1, we show that when $t \to \infty$, the competitive ratio of **RM**-Algorithm is at least $2n - 1$ even for the line metric and therefore a finite t is necessary for the input sensitive analysis.

In Section 4.2, for the line metric, we construct a set of initial server locations, a set of request locations and its arrival order so that the online matching produced by the **RM**-Algorithm is at least $\Omega(\log n)$ times the cost of an optimal offline matching. This lower bound suggests that the dependence of the input sensitive competitive ratio on $\log n$ may be necessary.

Finally in Section 4.3, we show that the competitive ratio of any online algorithm for an input instance $(\mathbb{M}, S)$ is at least $\Omega(\mu_{\mathbb{M}}(S'))$, where $S' = \arg\max_{S'' \subset S} \mu_{\mathbb{M}}(S'')$.

4.1 Choice of *t*

Let us assume that the set of servers S and requests R are points on a line where the locations of servers are as follows: $s_1 = -1$ and $s_i = (i-1)(1-\epsilon),$ for every $2 \leq i \leq n$ and $\epsilon \to 0$, i.e., $d(0, s_1) = 1; d(0, s_2) = d(s_i, s_{i+1}) = 1 - \epsilon,$ for every $2 \leq i \leq n-1$. The arrival order of requests is given by $r_1 = 0$ and $r_i = s_i = (i-1)(1-\epsilon),$ for every $2 \leq i \leq n$. The optimal matching M_{OPT} for the following configuration of servers and requests will be $(s_i, r_i),$ for every $1 \leq i \leq n$ and $w(M_{\text{OPT}}) = 1$. To compute the online matching M, when the request r_1 arrives, our algorithm

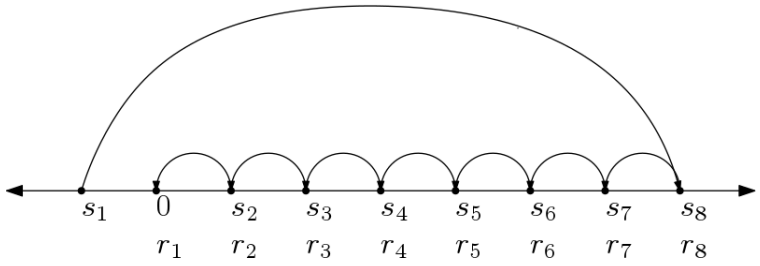

Figure 4.1. Example for proving the bound with larger value of *t*.

computes the minimum t-net-cost augmenting path from r_1 to s_2 and hence adds (s_2, r_1) to the matching after the first request r_1. If $t > f(\epsilon)$, where $f(\epsilon) = \frac{(n-2)(1-\epsilon)}{\epsilon}$, then for any request r_i, the respective minimum t-net-cost augmenting path P_i calculated by the algorithm will be from r_i to s_{i+1} with $\phi(P_i) = t(1-\epsilon),$ for every $2 \leq i \leq n-1$. The edges in the online matching are therefore, $(s_{i+1}, r_i),$ for every $1 \leq i \leq n-1$ and (r_n, s_1) and $w(M) \approx 2n-1$ according to the construction and as shown in Figure 4.1. The competitive ratio α in this example is $\alpha = w(M)/w(M_{\text{OPT}}) = 2n-1$. This shows that for a large value of t, the algorithm does not perform exceptionally well and hence the value of t has to be chosen carefully in order to achieve a better competitive ratio.

4.2 Optimal Analysis of RM-Algorithm

In this section, we show that the **RM**-Algorithm cannot achieve a competitive ratio better than $\Omega(\log n)$ for the line metric. Let us assume that the locations of the servers are $s_i = 2i - 1$ and the

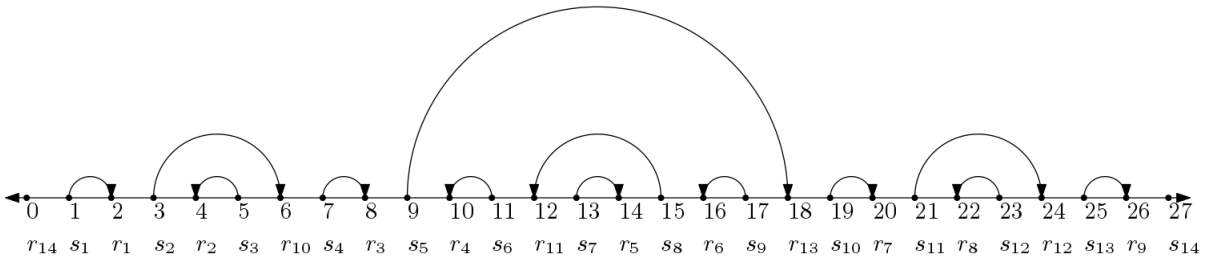

Figure 4.2. Example for proving the $\Omega(\log n)$ bound of **RM**-Algorithm.

locations of the requests are $r_i = 2i - 2$ for every $1 \leq i \leq n$. The weight of optimal matching M_{OPT} in this setting is $w(M_{\text{OPT}}) = n$ with the edges of the matching being (s_i, r_i) for every $1 \leq i \leq n$. We describe the first $2n/3$ requests all of which match at a cost 1. Let $r_1 = 2$, $r_i = r_{i-1} + 4$ if i is even and $r_i = r_{i-1} + 2$ if i is odd. When r_i arrives, the choices of the algorithm guarantee that both adjacent servers are present and form the minimum t-net cost paths. The algorithm chooses the server at location $r_i - 1$ if i is even, otherwise it chooses the server at $r_i + 1$ when i is odd. After processing these $2n/3$ requests, we incur a cost of $2n/3$ and have $n/3$ servers and requests each at a distance 3 from both the server on its left and right. Therefore the remaining $n/3$ servers and requests can be seen as a scaled (by 3) version of the initial input with fewer servers and requests. We repeat the same construction and match $2n/9$ requests at a cost of 3 each. We define requests to be classified in different levels where every request which matched with the servers with same edge cost are classified into same level i.e., first $2n/3$ fraction of requests are level 1 requests, next $2n/3$ fraction of requests are level 2 requests and so on.

In the end, we have for every level j, $j \geq 0$, a total of $(\frac{1}{3})^j n$ servers and requests. We match $\frac{2}{3}(\frac{1}{3})^j n$ of these requests to servers and incur a cost of 3^j for each request. Therefore the total cost for each level j is $\frac{2}{3}(\frac{1}{3})^j n \times 3^j = \frac{2n}{3}$. There are $\Omega(\log_3 n)$ levels and therefore the total cost of the online matching is $\Omega(n \log n)$ or $\frac{w(M)}{w(M_{\text{OPT}})} = \Omega(\log n)$.

4.3 Lower Bound for Online Bipartite Matching

The following section shows that given any metric space $\mathbb{M}$ and the initial server configurations S, any online algorithm A has a competitive ratio of at least $O(\mu_{\mathbb{M}}(S'))$, where $S' \subseteq S$ such that $S' = \arg\max_{S'' \subseteq S} \mu_{\mathbb{M}}(S'')$.

Let the set of server locations be $S = \{s_1, \ldots, s_n\}$ and the set of request locations be $R = \{r_1, \ldots, r_n\}$. Let us assume that $S' = \{s'_1, \ldots, s'_{n'}\}$. For servers in the set $S \setminus S'$, we place the requests at their locations so that they match to these servers at a cost of zero. If this is not the case, then the cost of online matching will only increase. Next, for the set S', we generate r'_1 at location s'_1. Then, the algorithm A assigns s'_1 to r'_1. For $2 \leq i \leq n'$, we generate r'_i at the location of the server s'_{i-1}. When all the requests have arrived, the online matching M consists of edges $(r'_1, s'_1), (r'_2, s'_2), \ldots, (r'_n, s'_{n'})$. Since $r'_i = s'_{i-1}$, the edges of M can be defined as $(s'_1, s'_2), \ldots, (s'_{n'-1}, s'_{n'})$ which is equal to the cost of the path $\langle s'_1, s'_2, \ldots, s'_{n'} \rangle$. The optimal

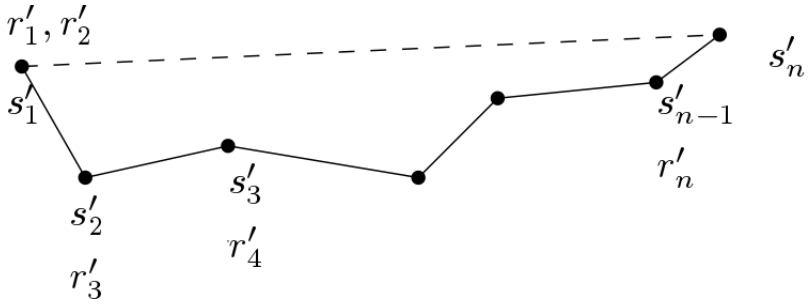

Figure 4.3. The online matching edges(denoted by the solid lines) and the optimal matching edge(denoted by dashed line) for the subset S' as per the lower bound construction.

matching M_{OPT} will match r'_i to s'_{i-1} for every $2 \leq i \leq n$ with 0-cost edges and $r'_1 = s_1$ to s_n.

Therefore, the competitive ratio is given by:

$$\begin{aligned}
\frac{w(M)}{w(M_{\text{OPT}})} &\geq \frac{\sum_{i=1}^{n'} d(s'_i, s'_{i+1})}{d(s'_1, s'_{n'})} \\
&\geq \frac{(\sum_{i=1}^{n'} d(s'_i, s'_{i+1})) + d(s'_1, s'_{n'})}{2d(s'_1, s'_{n'})} \\
&\geq \frac{\text{TSP}(S')}{2\text{DIAM}(S')} \\
&= \Omega(\mu(S')).
\end{aligned}$$

Chapter 5

Conclusions and Future Work

In this thesis, we provided a novel input sensitive analysis of the **RM**-Algorithm in the adversarial model. For a given metric space $\mathbb{M}$, we showed that this algorithm achieves a competitive ratio of $O(\mu_{\mathbb{M}}(S')\log^2 n)$, where $S' \subseteq S$. While the **RM**-Algorithm is oblivious to the underlying metric space $\mathbb{M}$, it nevertheless performs near-optimally under $\mathbb{M}$. It is possible to construct a 1-dimensional example for which the competitive ratio of this algorithm is $\Omega(\log n)$. Therefore, this algorithm cannot simultaneously achieve a competitive ratio of $\Theta(\mu_{\mathbb{M}}(S'))$ for the line metric. We conclude by stating the following open problem.

- Is there an input-sensitive online algorithm for bipartite matching that achieves a competitive ratio of $\Theta(\mu_{\mathbb{M}}(S'))$ for any metric space?
- For Random Arrival Model, is it possible to show an input sensitive analysis for the **RM**-Algorithm?

Glossary

adversarial model Adversary generates a request sequence to maximize the competitive ratio. 2

alternating path Path whose edges alternate between the edges in the matching and those not in the matching. 9

augmenting path Alternating path which starts at a free request and ends a at free server. 9

bipartite graph Consists of set of vertices which can be partitioned into two vertex sets such that edges link vertices from one vertex set to the other. 3

clusters Requests from each inner group are divided into clusters based on their order of arrival and distance from other clusters. 19

free vertex Vertex which is not matched in the current matching. 9

inner groups Requests are partitioned into $O(\log n)$ inner groups based on their minimum t-net cost augmenting paths. 18

matching Vertex-disjoint set of edges in a bipartite graph. 3

oblivious adversary model Adversary generates the the request locations and the arrival order but does not know the random choices of the algorithm. 2

optimal matching Minimum-cost perfect matching. 4

optimal path Augmenting path formed for each request by taking the symmetric difference of the optimal matching to the offline matching with respect to the representative request of its cluster. 21

outer groups Requests are partitioned into $O(\log n)$ outer groups based on the distance of the farthest vertex on their optimal paths from the representative request of their clusters. 19

random arrival model Adversary generates the request locations but not the arrival order. 2

representative request It is the centre of the cluster and the first request which came in the cluster. 20

Symbols

C_k^i Represents cluster k in inner group i. 20

$M(i,k)$ Offline matching maintained by the algorithm right before $rc(i,k)$. 21

Q_g Represents the optimal path for any request r_g. 21

$R(i,j)$ Set of requests that have inner group i and outer group j. 21

$R(i,j,k)$ Requests of $R(i,j)$ that belong to cluster k. 21

$\mathcal{IB}(i,j,k)$ Inner ball for the set $R(i,j,k)$. 22

$\mathcal{OB}(i,j,k)$ Outer ball for the set $R(i,j,k)$. 22

$\beta(i,j,k)$ $\beta(i,j,k) = |R(i,j,k)|$. 21

δ_g For $r_g \in C_k^i$, distance between $rc(i,k)$ and the farthest vertex on Q_g. 21

$\mathcal{IG}_i$ Represents inner group i. 20

$\mu_{\mathbb{M}}(W)$ Stretch of a point set W in the metric space $\mathbb{M}$. 17

ϕ_i t-net cost of the augmenting path computed by the algorithm for request r_i=$\phi_t(P_i)$. 12

$\mathbf{r}_{\mathcal{IB}}(i)$ Inner radius for inner group i. 22

$\mathbf{r}_{\mathcal{OB}}(j)$ Outer radius for outer group i. 22

$\mathcal{OG}_i$ Represents outer group i. 21

n_k^i Number of requests in cluster C_k^i. 20

$rc(i,k)$ Representative request for cluster k in inner group i. 21

$y(\cdot)$ Dual weight of any vertex. 10

$M_{\mathbf{OPT}}$ Optimal Matching. 4

Bibliography

[1] Antonios Antoniadis, Neal Barcelo, Michael Nugent, Kirk Pruhs, and Michele Scquizzato. A o(n) -competitive deterministic algorithm for online matching on a line. In *Approximation and Online Algorithms - 12th International Workshop, WAOA 2014, Wrocław, Poland, September 11-12, 2014, Papers*, pages 11–22, 2014.

[2] N. Bansal, N. Buchbinder, A Madry, and J. Naor. A polylogarithmic-competitive algorithm for the k-server problem. In *Proceedings of the IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 267–276, Oct 2011.

[3] Nikhil Bansal, Niv Buchbinder, Anupam Gupta, and Joseph Naor. An $O\left(\log^2 k\right)$-competitive algorithm for metric bipartite matching. In *Algorithms - ESA 2007, 15th Annual European Symposium, Eilat, Israel, October 8-10, 2007, Proceedings*, pages 522–533, 2007.

[4] A. Gupta and K. Lewi. The online metric matching problem for doubling metrics. In *Automata, Languages, and Programming*, volume 7391, pages 424–435. 2012.

[5] Bala Kalyanasundaram and Kirk Pruhs. Online weighted matching. *Journal of Algorithms*, 14(3):478–488, 1993.

[6] Chinmay Karande, Aranyak Mehta, and Pushkar Tripathi. Online bipartite matching with unknown distributions. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 587–596, New York, NY, USA, 2011. ACM.

[7] Samir Khuller, Stephen G. Mitchell, and Vijay V. Vazirani. On-line algorithms for weighted bipartite matching and stable marriages. *Theoretical Computer Science*, 127(2):255–267, 1994.

[8] Elias Koutsoupias and Akash Nanavati. *Approximation and Online Algorithms: First International Workshop, WAOA 2003, Budapest, Hungary, September 16-18, 2003. Revised Papers*, chapter The Online Matching Problem on a Line, pages 179–191. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[9] Elias Koutsoupias and Christos H. Papadimitriou. On the k-server conjecture. *J. ACM*, 42(5):971–983, September 1995.

[10] M. Mahdian and Q. Yan. Online bipartite matching with random arrivals: An approach based on strongly factor-revealing lps. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*, STOC '11, pages 597–606, 2011.

[11] Mark S. Manasse, Lyle A. McGeoch, and Daniel D. Sleator. Competitive algorithms for server problems. *J. Algorithms*, 11(2):208–230, May 1990.

[12] Sharath Raghvendra. A robust and optimal online algorithm for minimum metric bipartite matching. In Klaus Jansen, Claire Mathieu, José D. P. Rolim, and Chris Umans, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2016)*, volume 60 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:16, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[13] Giuseppe Vitali. Sui gruppi di punti e sulle funzioni di variabili reali (in italian); (title translation) on groups of points and functions of real variables. *Atti dell'Accademia delle Scienze di Torino JFM 39.0101.05*, 43:75–92, 1908.