Online Optimization in Routing and Scheduling

by

Michael Robert Wagner

M.Eng. in Electrical Engineering and Computer Science
Massachusetts Institute of Technology (2001)

B.S. in Electrical Engineering and Computer Science
Massachusetts Institute of Technology (2000)

B.S. in Mathematics
Massachusetts Institute of Technology (2000)

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2006

Author .....
Sloan School of Management
May 18th, 2006

Certified by .....
Patrick Jaillet
Edmund K. Turner Professor
Thesis Supervisor

Accepted by .....
James B. Orlin
Edward Pennell Brooks Professor of Operations Research
Co-director, Operations Research Center

Online Optimization in Routing and Scheduling

by

Michael Robert Wagner

Submitted to the Sloan School of Management
on May 18th, 2006, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Operations Research

Abstract

In this thesis we study online optimization problems in routing and scheduling. An online problem is one where the problem instance is revealed incrementally. Decisions can (and sometimes must) be made before all information is available. We design and analyze (polynomial-time) online algorithms for a variety of problems. We utilize worst-case competitive ratio (and relaxations thereof), asymptotic and Monte Carlo simulation analyses in our study of these algorithms.

The focus of this thesis is on online routing problems in arbitrary metric spaces. We begin our study with online versions of the Traveling Salesman Problem (TSP) and the Traveling Repairman Problem (TRP). We then generalize these basic problems to allow for precedence constraints, capacity constraints and multiple vehicles. We give the first competitive ratio results for many new online routing problems. We then consider resource augmentation, where we give the online algorithm additional resources: faster servers, larger capacities, more servers, less restrictive constraints and advanced information. We derive new worst-case bounds that are relaxations of the competitive ratio. We also study the (stochastic) asymptotic properties of these algorithms – introducing stochastic structure to the problem data, unknown and unused by the online algorithm. In a variety of situations we show that many online routing algorithms are (quickly) asymptotically optimal, almost surely, and we characterize the rates of convergence.

We also study classic machine sequencing problems in an online setting. Specifically, we look at deterministic and randomized algorithms for the problems of scheduling jobs with release dates on single and parallel machines, with and without preemption, to minimize the sum of weighted completion times. We derive improved competitive ratio bounds and we show that many well-known machine scheduling algorithms are almost surely asymptotically optimal under general stochastic assumptions.

For both routing and sequencing problems, we complement these theoretical derivations with Monte Carlo simulation results.

Thesis Supervisor: Patrick Jaillet
Title: Edmund K. Turner Professor

Acknowledgments

This dissertation represents the final chapter of my decade-long education at MIT and there are many people to thank. I will do my best to acknowledge everyone.

First and foremost, I would like to thank my thesis advisor Patrick Jaillet. I am deeply indebted to Patrick for his research guidance, generous support and true interest in my well-being. I could not have had a better advisor. To be truthful, words are not sufficient to express my gratitude. I am also grateful to the other members of my thesis committee, Michel Goemans and Jim Orlin, for their help and guidance over the years in academic as well as other affairs. I thank Andreas Schulz for numerous occasions where he gave me great advice. I am also thankful for the wonderful staff at the ORC – many thanks are due to Paulette and Laura for the countless favors over the years. Thanks to Andrew and Veronica are also due. I also thank Vladimir Barzov for implementing many of the algorithms in this thesis as well as for collaborating on a couple of theoretical problems.

Of course, the creation of this thesis has been supported by the friendship of many people over the years. In the ORC, I must especially thank Margrét (I couldn't have wished for a better officemate), Katy (fellow Course 6 alum), Juliane (aka Blondie/Cookie) and Mike M (fellow ex-frat boy) for their true friendship and support over the last few years. Also, commiseration with Guillaume, Alex B. and Elodie made the academic recruiting process bearable. Carol, Lincoln and Yann always contributed to making the ORC a fun and social place and I am grateful. Finally, I must not forget those who passed out of this place before me: special thanks to Agustin, Nico and José (aka Mula) for their friendship and advice. Special thanks are due to José for our fruitful research collaboration (which appears as Chapter 3 in this thesis).

Friends outside of the ORC have also been a great source of strength throughout the years. Ever since being undergrads together at MIT, I have truly valued the friendship of Rita (the sister I never had), Dave (my academic partner in crime) and Ramy (my Buddha advisor). Crystal, who escaped the East Coast years before me,

has been there for me through thick and thin and I am grateful for her support and friendship. Thanks are due to Alex V. for always hosting me during my NYC jaunts and for being a true friend. I thank Sandra, my Californian friend in Massachusetts, for her friendship and support during my last couple of years on the East Coast. Finally, I must mention and thank the Slack Pack, particularly Kup, Drunkard and Big Daddy; these slackers were always there for me when I needed them.

Finally, there is family. This thesis would not have been possible without the support of my family. First, I must thank my grandfather, who first told me of a magical place called MIT a lifetime ago. My entire Peruvian family has always been behind me 100% and I am eternally grateful to them. I thank Maximina for her support and excellent meals. I thank my Dad for always being there with a ready ear. Finally, I thank my Mom, whose support and love always rose above the rest.

Cambridge, May 2006

Michael R. Wagner

Contents

List of Figures

Chapter 1

Introduction

The Traveling Salesman Problem (TSP) is one of the most intensely studied problems in optimization. In one of its simplest forms we are given a metric space and a set of points in the space, representing cities. Given an origin city, the task is to find a tour of minimum total length, beginning and ending at the origin, that traverses each city at least once. Assuming a constant speed, we can interpret this objective as minimizing the time required to complete a tour.

The TSP is relevant in many settings. Usually, but not exclusively, the TSP is useful in the solution of transportation problems, such as designing routes for buses, the everyday routing of pickup/delivery trucks, designing transportation networks, etc. Less obvious applications include Very Large Scale Integration (VLSI) design and genome sequencing. The TSP is also central to the solution of more complicated problems, such as fleet organization and routing, capacitated routing and many other routing problems with various constraints.

The assumption that problem instances are completely known a priori is unrealistic in many applications. Taxi services, buses and courier services, for example, require an *online* model in which cities (points) to be visited are revealed over time, while the server is en route serving previously released requests. Therefore, we are motivated to consider online versions of the TSP. To make this notion more precise, we return to the TSP definition where we minimize the time needed to visit all cities and return to the origin. We also add the restriction that each city must be visited on or after

its *release date*; in this case the problem is known as the "TSP with release dates." The online TSP is simple: Each city is only revealed at its release date, independent of the salesman's location. Before a given city's release date, the online algorithm is not aware that the city exists; furthermore, the number of cities is not known a priori either. Note that, in general, we do not introduce any stochastic information; there are no distributions nor forecasts for the city locations, release dates or even the number of cities.

The focus of this thesis is on designing and studying algorithms for various online problems in routing and scheduling. We analyze algorithms with and without computational efficiency requirements. In addition to the online TSP, we also study other online routing problems, such as the online Traveling Repairman Problem (TRP), where the objective is to minimize the weighted sum of times that cities are first visited, the online k-TSP, where the server must pick a subset of cities of size k to visit, the online TSP with multiple vehicles, the online TSP with precedence and capacity constraints, the online TRP with precedence constraints and several hybrids of these problems. In most cases, we study the situation where city locations are elements of a generic metric space of arbitrary dimension. We also study online machine scheduling, focusing on multiple machine scheduling with the objective of minimizing the weighted sum of job completion times.

We evaluate algorithms for these online problems using a variety of methods. We use the *competitive ratio*, which is defined as the worst case ratio of the online algorithm's cost to the cost of an optimal offline algorithm. An offline algorithm is one that is provided with all the problem data a priori. We also consider resource augmentation (only for online routing), where we give the online algorithm additional resources. The motivation for resource augmentation is to preclude the worst-case instances that induce unrealistic competitive ratios; in this way, the competitive ratios we derive are more practical and meaningful. We quantify the improvement in competitive ratio of giving the online algorithm additional resources, such as (a) advanced information, (b) faster speeds, (c) additional salesmen/servers, (d) larger capacities and other resource improvements. Finally, we also use the asymptotic competitive

ratio, which is loosely defined as a competitive ratio for "large" problem instances; using this approach, we are able to show that, under certain conditions, many online algorithms are (quickly) asymptotically optimal.

Outline: The rest of this chapter is organized as follows: In Section 1.1, we further motivate our study of online optimization by discussing a number of applications. In Section 1.2 we give the general definitions and terminology used in online optimization. In Section 1.3, we give a literature review of problems in online routing and machine scheduling. Finally, in Section 1.4, we summarize the contributions of this thesis.

1.1 Applications of Online Optimization

In this section, we discuss a number of applications of online optimization and the related discipline of real-time optimization. Real-time optimization essentially studies the same type of problems that are studied in online optimization, but from a less theoretical point of view. Essentially, real-time optimization solves problems where the problem instance is again revealed incrementally, but the focus is on (fast) practical algorithms and heuristics that solve quickly and give good solutions, but without theoretical performance guarantees.

We first discuss applications in routing and scheduling. FedEx and other courier companies offer many real-time services. For example, FedEx provides same day pickup and delivery. In many cases, there exist data for customers that request these services and forecasts can be created to help in designing routes. However, in the case where FedEx is beginning service in a new area, online and real-time optimization is a more appropriate approach until data can be gathered about the customer base. Another application is in the real-time fleet management where real-time information is used to manage a fleet of vehicles; see Yang, Jaillet and Mahmassani [71]. A third example comes from a rather new industry: Jet taxis. A jet taxi is a small jet that serves regional airports; customers can request transportation from one airport to another in real-time. Since this is a new industry, data is not yet available to

create forecasts of customer demand, so online and real-time optimization is again appropriate for analyzing these problems. Management of a fleet of taxis can also benefit from online analysis. Considering online machine scheduling, there are many applications in manufacturing. For a very interesting example, consider the case study presented in Ascheuer, Grotschel and Rambau [7]. Applications in warehousing also exist: consider the real-time orders that amazon.com receives and must fill quickly (e.g., customers requesting next day delivery).

There are also a number of applications of online optimization in computer science. For example, paging in a memory system is one of the first applications of online optimization. Searching an unknown domain for a prize is also popular (e.g., see Jaillet and Stafford [42]); this latter problem has applications in robotics. Routing and call admission in communications networks are also popular areas of research in online optimization.

1.2 Online Optimization Problems and Algorithms

1.2.1 Definitions

An online optimization problem is one where the problem instance is revealed incrementally. Decisions can or must be made before the entire problem instance is revealed. Any algorithm that gives a solution to an online problem is denoted an online algorithm.

Let I denote a complete instance of an online optimization problem. We partition I into a sequence of requests σ_i: $I = (\sigma_1, \sigma_2, \ldots, \sigma_n)$. These requests are revealed incrementally and collectively define the problem instance. In general, the number of requests n is not known a priori. There are two main frameworks for revealing these requests: sequentially and dynamically.

In the sequential model, when σ_i is revealed, a decision by the online algorithm must be made before σ_{i+1} is revealed. The well-known online k-server problem (see Borodin and El-Yaniv [18]) is of the sequential type.

Online problems under the dynamic model usually have a time component. In this model, the requests are revealed dynamically over time, irrespective of the actions of the online algorithm. The time that a request is revealed is denoted the request's *release date.* In this thesis, we exclusively use the dynamic online model.

Definitions Applied to Online Routing

We briefly indicate how the above general definition of online optimization fits into our study of online routing problems. Each request σ_i is revealed to the online algorithm at the request's release date. Additionally, each σ_i will consist of a set of cities and possibly some extra data (e.g., city weights for the online Traveling Repairman Problem). Note that if each σ_i consists of a single city, we have the online TSP. More formal problem definitions will be found in the subsequent chapters.

Finally, a note on terminology. "Cities," "points," and "locations" are used interchangeably, as are "Salesman," "server" and "vehicle" (though we attempt to adhere to common usage based on the underlying problem definition).

Definitions Applied to Online Machine Scheduling

We briefly indicate how the above general definition of online optimization fits into our study of online machine scheduling problems. Each request σ_i is revealed to the online algorithm at the request's release date. Each σ_i will correspond to a job and will consist of the job's weight and processing time. Again, more formal problem definitions will be found in the subsequent chapters.

1.2.2 Competitive Analysis

We now describe a framework for evaluating the quality of online algorithms. For simplicity, we consider only minimization problems. Central concepts in the analysis of online algorithms are the notions of competitiveness and the competitive ratio. An online algorithm is said to be r-competitive ($r \geq 1$) if, given any instance of the problem, the cost of the solution given by the online algorithm is no more than r

multiplied by that of an optimal offline algorithm:

$$\text{Cost}_{\text{online}}(I) \leq r\text{Cost}_{\text{optimal}}(I), \ \forall \text{ problem instances } I.$$

The infimum over all r such that an online algorithm is r-competitive is called the competitive ratio of the online algorithm. An online algorithm is said to be best-possible if there does not exist another online algorithm with a strictly smaller competitive ratio.

We may also define asymptotic competitiveness when there is a natural notion to the size of the problem. Let $n \in \mathbb{Z}^+$ index the size of the problem instance; symbolically, I_n is a problem instance of size n. We can say that an online algorithm is asymptotically r-competitive if there exists a $n_0 > 0$ such that for all $n \geq n_0$, $\text{Cost}_{\text{online}}(I_n) \leq r\text{Cost}_{\text{optimal}}(I_n), \ \forall$ problem instances I_n.

We note that the competitive ratio is a worst case bound and some say it is too pessimistic a measure for evaluating online algorithms. Throughout this thesis, we provide relaxations of the competitive ratio for evaluating our online algorithms.

Randomized Online Algorithms

In this thesis, we also consider randomized online algorithms. A randomized algorithm is a probabilistic distribution over a set of deterministic algorithms. Two runs of a randomized algorithm on the same instance can result in two different outputs.

Competitive analysis considers other factors in the context of randomized online algorithms. Implicit in the definition of the deterministic competitive ratio was the notion of an offline *adversary* that gives the online algorithm a worst possible instance. In other words, an offline adversary knows all details of an online algorithm and gives an instance that induces the competitive ratio. When dealing with randomized online algorithms, the definitions of the competitive ratio and competitiveness depend on how much the adversary knows about the randomization. In all cases, we now compare the online cost with an *adversarial* cost.

We introduce three adversaries that are used in the analysis of randomized on-

line algorithms. These adversaries are denoted the oblivious, adaptive-online and adaptive-offline adversaries. We exclusively utilize the oblivious adversary in this thesis; we define the other two adversaries for completeness.

The *oblivious* adversary knows the randomized online algorithm as well as the distribution that underlies the randomness. However, the oblivious adversary has no information about the *realizations* of the randomness underlying the randomized algorithm. Therefore, the oblivious adversary creates a problem instance a priori without seeing any of the actions of the online algorithm. The adversarial cost is deterministic and equal to the optimal offline cost. In this case, in the definition of the competitive ratio, we simply replace the cost of the online algorithm with its expected cost:

$$\mathbb{E}\left[\text{Cost}_{\text{online}}(I)\right] \leq r\text{Cost}_{\text{optimal}}(I),\ \forall \text{ problem instances } I.$$

The *adaptive-online* adversary knows all probabilistic realizations of the randomized online algorithm *up to the current time.* In this case, the adaptive-online adversary creates the problem instance in real time and must also serve the created instance in real time. Therefore, the probabilistic adversarial cost is calculated online and is not necessarily the optimal offline cost on the resulting instance. The definition of the competitive ratio is modified as follows:

$$\mathbb{E}\left[\text{Cost}_{\text{online}}(I) - r\text{Cost}_{\text{adaptive-online}}(I)\right] \leq 0,\ \forall \text{ problem instances } I;$$

note that we use $\text{Cost}_{\text{adaptive-online}}(I)$ rather than $\text{Cost}_{\text{offline}}(I)$ in the definition of the competitive ratio for the adaptive-online adversary.

The *adaptive-offline* adversary again knows all probabilistic realizations of the randomized online algorithm *up to the current time* and creates the problem instance in real time. However, the probabilistic adversarial cost is the optimal offline cost. The definition of the competitive ratio is modified as follows:

$$\mathbb{E}\left[\text{Cost}_{\text{online}}(I) - r\text{Cost}_{\text{offline}}(I)\right] \leq 0,\ \forall \text{ problem instances } I.$$

We conclude this section with some standard implications taken from [18]. If $c_{oblivious}$, $c_{adaptive-online}$ and $c_{adaptive-offline}$ are the competitive ratios for a given randomized online algorithm against oblivious, adaptive-online and adaptive-offline adversaries, respectively, then

$$c_{oblivious} \leq c_{adaptive-online} \leq c_{adaptive-offline}.$$

Futhermore, we have the following results.

Theorem 1 ([18]) *If there exists a c-competitive randomized online algorithm against an adaptive-offline adversary, then there also exists a deterministic c-competitive online algorithm.*

Theorem 1 states that randomization does not help to improve competitive ratios with respect to an adaptive-offline adversary.

Theorem 2 ([18]) *If there exist a c_1-competitive randomized online algorithm against an adaptive-online adversary and a c_2-competitive randomized online algorithm against an oblivious adversary, then there also exists a randomized online algorithm that is c_1c_2-competitive against an adaptive-offline adversary.*

1.2.3 Comparision with Complexity Theory

The study of online algorithms has many similarities to the study of approximation algorithms. The design of online algorithms addresses the situation where there is a limitation on information. Approximation algorithm design addresses the situation where there is a limitation on computational power. In both cases, a worst case measure is usually applied: the competitive ratio and the approximation ratio, respectively.

In this thesis, we consider both pure online problems, where we are not concerned with computational issues, as well as problems where we are concerned with both the lack of information and the lack of computational power. In this latter case, our competitive ratios are also approximation ratios.

1.3 Literature Review

1.3.1 Routing Literature Review

The literature for the TSP is vast. The interested reader is referred to the books by Lawler, Lenstra, Rinnooy Kan, Shmoys [53] and Korte and Vygen [49] for comprehensive coverage of results concerning the TSP. Probabilistic versions of the TSP, where a different approach is used to represent limited knowledge of the problem instance, have also attracted interest (e.g., see Jaillet [41] and Bertsimas [14]). Offline routing problems with release dates can be found in Psaraftis, Solomon, Magnanti, Kim [60] and Tsitsiklis [69]. We also mention two offline results that will play a part in our analysis: the 3/2-approximation algorithm for the TSP in metric space by Christofides [25] and the polynomial-time approximation scheme for the TSP in Euclidean space by Arora [6].

A systematic study of online algorithms was given by Sleator and Tarjan [67], who suggested comparing an online algorithm with an optimal offline algorithm. Karlin, Manasse, Rudolph, Sleator [48] introduced the notion of a *competitive ratio*. Online algorithms have been used to analyze paging in computer memory systems, distributed data management, navigation problems in robotics, multiprocessor scheduling, etc.; see the books of Borodin and El-Yaniv [18] and Fiat and Woeginger [29] for more details and references.

Research concerning online versions of the TSP and TRP have been introduced relatively recently. Kalyanasundaram and Pruhs [44] have examined a unique version of an online traveling salesman problem where new cities are revealed locally during the traversal of a tour (i.e., an arrival at a city reveals any adjacent cities that must also be visited). More related to our thesis is the stream of works which started with the paper by Ausiello, Feuerstein, Leonardi, Stougie, Talamo [10]. In this paper, the authors studied the online TSP version we consider here; they analyzed the problem on the real line and on general metric spaces, developing online algorithms for both cases and achieving a best-possible online algorithm for general metric spaces, with a competitive ratio of 2. These authors also provide a polynomial-time online

algorithm, for general metric spaces, which is 3-competitive. Subsequently, Ausiello, Demange, Laura, Paschos [9] gave a polynomial-time algorithm, for general metric spaces, which is 2.78-competitive. Jaillet and Wagner [43] gave a $(2+\epsilon)$-competitive polynomial-time algorithm for Euclidean spaces with dimension $d \geq 2$. Lipmann [54] developed a best-possible online algorithm for the real line, with a competitive ratio of approximately 1.64. Blom, Krumke, de Paepe, Stougie [15] gave a best-possible online algorithm for the non-negative real line, with a competitive ratio of $\frac{3}{2}$. This last paper also considers different adversarial algorithms in the definition of the competitive ratio.

Considering the online TRP, Feuerstein and Stougie [28] gave a lower bound of $(1+\sqrt{2})$ for the competitive ratio and a 9-competitive algorithm, both for the online TRP on the real line. Krumke, de Paepe, Poensgen, Stougie [51] improved upon this result to give a $(1+\sqrt{2})^2$-competitive deterministic algorithm for the online TRP in general metric spaces as well as a Θ-competitive randomized algorithm, where $\Theta \approx 3.64$; in this thesis, we correct this result to $\Theta \approx 3.86$ (see also [52]).

Resource augmentation for online problems was introduced in [67]. These authors show that it is possible for an online paging algorithm to have a constant competitive ratio if it is given a constant fraction more cache locations than the offline algorithm. Server resource augmentation was considered by Young [72] for the k-server problem and Kalyanasundaram and Pruhs [45] for the online weighted matching problem. Kalyanasundaram and Pruhs [46] consider speed and processor augmentation in online machine scheduling. In [43], information augmentation is present in the form of *disclosure dates*; a similar approach was taken by Allulli, Ausiello, Laura [2] in the form of a *lookahead*. Ausiello, Allulli, Bonifaci, Laura [8] also consider the behavior of online routing algorithms as a function of the number of servers. Other frameworks for addressing the limitations of the competitive ratio have also been introduced; see Ben-David and Borodin [13], Koutsoupias and Papadimitriou [50] and Raghavan [61].

1.3.2 Machine Scheduling Literature Review

There has been an enormous amount of work on parallel machine scheduling. As we do not intend to do a complete review of results in the area, let us only mention some of the most relevant literature on online scheduling problems directly related to the matter of this thesis.

To the best of our knowledge, the first deterministic online algorithm for $P|r_j|\sum_j w_jC_j$ was given by Hall, Schulz, Shmoys and Wein [39]. They design a $(4+\varepsilon)$-competitive algorithm. Prior to this thesis, the best-known deterministic algorithms for both $P|r_j|\sum_j w_jC_j$ and $P|r_j, pmtn|\sum_j w_jC_j$ were recently given by Megow and Schulz [57] and are 3.28 and 2-competitive, respectively. They also show that the former algorithm has a competitive ratio between 2.78 and 3.28 while the latter analysis is tight.

Considering randomized algorithms, a $(2.89+\varepsilon)$-competitive algorithm for $P|r_j|\sum w_jC_j$, was obtained by Chakrabarti, Phillips, Schulz, Shmoys, Stein and Wein [20]. Schulz and Skutella [64] give randomized strategies that are 2-competitive for both $P|r_j, pmtn|\sum_j w_jC_j$ and $P|r_j|\sum_j w_jC_j$. Related results have been obtained by Chekuri, Motwani, Natarajan and Stein [22] and Phillips, Stein and Wein [58] when the objective is to minimize the average completion time of the schedule. In a more restricted setting, Chou, Queyranne and Simchi-Levi [24] consider the online $P|r_j|\sum_j w_jC_j$ with lower and upper bounds on jobs' weights and processing times; the authors prove that the online *weighted shortest processing time* heuristic is asymptotically optimal. They even extend this to the problem $Q|r_j|\sum_j w_jC_j$. Similar results can be found in the papers by Liu, Queyranne and Simchi-Levi [56], Chou, Liu, Queyranne and Simchi-Levi [23] and Kaminsky and Simchi-Levi [47].

We also mention some single machine scheduling results, as our work essentially extends these analyses to the parallel machine case. Using the idea of α-points and mean-busy-time relaxations, Goemans, Queyranne, Schulz, Skutella and Yang [35] designed a deterministic 2.4143-competitive and a randomized 1.6853-competitive algorithm for the online $1|r_j|\sum_j w_jC_j$. A similar approach was taken by Schulz and

Skutella [63] to give a randomized $\frac{4}{3}$-competitive algorithm for $1|r_j, pmtn|\sum_j w_j C_j$; Sitters [66] gave a 1.56-competitive deterministic algorithm for the same problem. On the other hand, Anderson and Potts [4] provide a best possible deterministic online algorithm for $1|r_j|\sum_j w_j C_j$ which has a competitive ratio of 2. Additionally, Savelsbergh, Uma and Wein [62] perform an extensive computational study of a number of heuristics and approximation algorithms for the offline single machine problem $1|r_j|\sum_j w_j C_j$. They conclude that LP-based approximation algorithms for this problem perform very well.

Let us now discuss some lower bounds on the competitive ratios for certain problems. Hoogeveen and Vestjens [40] showed that there is no deterministic algorithm with competitive ratio strictly better than 2 for $1|r_j|\sum_j w_j C_j$. On the other hand Stougie and Vestjens [68] showed that $\frac{e}{e-1}$ is a lower bound on the competitive ratio of online randomized algorithms for the same problem. In the parallel machine case, Vestjens [70] proved that any deterministic algorithm for $P|r_j|\sum_j w_j C_j$ (resp. $P|r_j, pmtn|\sum_j w_j C_j$) has a competitive ratio of at least 1.309 (resp. $\frac{22}{21}$). Seiden [65] proved that any randomized algorithm for $P|r_j|\sum_j w_j C_j$ has a competitive ratio of at least 1.157. To the best of our knowledge, there are no specific lower bounds for randomized algorithms for $P|r_j, pmtn|\sum_j w_j C_j$.

Finally, we remark that all the machine scheduling problems considered in this thesis admit polynomial time approximation schemes (offline) [1].

1.4 Thesis Outline and Contributions

1.4.1 Chapter 2: Online Routing Optimization Problems

In this chapter, we study a variety of online routing problems; we evaluate algorithms using the classic competitive ratio and competitiveness notions. We first consider the online TSP and present a best-possible online algorithm, with a competitive ratio of 2, given in [10]. We then give a polynomial-time generalization that is 2ρ-competitive, where ρ is an approximation ratio for the TSP with no release dates.

We then introduce precedence and capacity constraints to the online TSP. We design an online algorithm that has a competitive ratio of at most 2; the power of this statement is that adding precedence and capacity constraints to the online TSP does not increase the competitive ratio. Considering polynomial-time algorithms, a modification to our algorithm is 2ρ-competitive, where ρ is an approximation ratio for a simpler offline problem.

Next, we study multiple server routing problems (without precedence and capacity constraints) and show results nearly identical to those just mentioned: We design new algorithms with competitive ratios of 2 and at most 2ρ. Adding servers to the problem statement does not increase nor decrease the competitive ratio with respect to the online TSP.

We also present existing results for the online TRP and the online k-TSP from the literature. For the online TRP, we present the $(1+\sqrt{2})^2$-competitive algorithm given in [51] and for the online k-TSP, we present the 2-competitive algorithm given in [9]. These results are presented here for comparison because we later generalize them in our study of resource augmentation.

Research in this chapter is joint work with Patrick Jaillet.

1.4.2 Chapter 3: Online Machine Scheduling Problems

In this chapter, we study online machine scheduling problems; we again evaluate algorithms using the classic competitive ratio and competitiveness notions. We exclusively study parallel machine online scheduling, where we have m identical machines on which to schedule n jobs. We study the minsum objective, where we minimize the weighted sum of job completion times. We consider both preemptive and non-preemptive situations. We study both deterministic and randomized algorithms. We give improved competitive ratio bounds for all problems considered.

We generalize the ideas given in [35] for the single machine case to the parallel machine problem $P|r_j|\sum_j w_j C_j$. As in that paper, our algorithm simulates a preemptive single machine scheduling problem on a virtual machine that is m times faster (where m is the number of machines). As soon as a fraction α of a given job has been

processed in the fast machine, the algorithm will put such a job in a FIFO queue and will schedule it in the parallel machines at the first point in time at which a machine is idle and all jobs with higher priority have been assigned. By choosing an appropriate value of α, namely $\alpha = (\sqrt{5}-1)/2$, we can prove that our algorithm is 2.618-competitive, improving upon the 3.28-competitive algorithm given in [57]. Our algorithm is deterministic and works online.

As in [35] we will show that the algorithm just described can be improved with the help of randomization. Basically, instead of taking a fixed value of alpha for all jobs, we can choose different α_j's for different jobs, and moreover, choose these values at random according to a given distribution. We give a randomized ϱ_m-competitive online algorithm for $P|r_j|\sum_j w_j C_j$, where $\varrho_m < 2$ for all $m \geq 1$. Here, m denotes the number of machines and ϱ_m is obtained implicitly. Our result improves upon the 2-competitive randomized algorithms in [64]. In contrast to their work our algorithm has the desirable property of being a list-scheduling algorithm and uses only one step of randomization. The algorithm we present can be seen as the parallel machine extension of the algorithm in [35] for a single machine. Indeed, the competitive ratio that it achieves is 1.6853 for $m = 1$ (as in Goemans et al.); for $m = 2, 3$ and 4 it is 1.8382, 1.8915 and 1.9184, respectively.

Following the algorithmic idea above, we present a randomized ρ_m-competitive online algorithm for $P|r_j, pmtn|\sum_j w_j C_j$, where $\rho_m < 2$ for all $m \geq 1$. Again, m denotes the number of machines and ρ_m is obtained implicitly. The reader may wish to compare our result with the current best algorithm to date: the deterministic algorithm in [57], which has a competitive ratio of 2 (and not better than 2) for any number of machines. Additionally, our algorithm can be simultaneously seen as an extension of the result in [57] and of the result in [63]. Indeed for a single machine, the competitive ratio of our algorithm is 4/3, as in [63]; for two, three and four machines it is 1.3867, 1.603 and 1.7173, respectively. In general we have that $\rho_m < 2 - 1/m$, for $m > 1$.

Research in this chapter is joint work with José Correa.

1.4.3 Chapter 4: Resource Augmentation in Online Routing

In this chapter, we study resource augmentation for online routing problems. We give the online algorithm additional resources and we quantify the improvement in the competitive ratio. We study advanced information, speed augmentation, vehicle augmentation and constraint augmentation. In Chapter 5, we study capacity augmentation under additional assumptions.

We introduce the notion of "disclosure dates," i.e., dates at which requests become known to the online algorithm, ahead of the release dates (the dates at which requests can first be served). In addition to providing more realism, the introduction of this advanced information is a natural mechanism to increase the "power" of online players against all-knowing adversaries in a competitive analysis framework. Note, also, that these disclosure dates provide a natural bridge between online routing problems and their offline versions – when all the disclosure dates are zero, we have the offline problems; when all the disclosure dates are equal to their respective release dates, we have the online routing problems considered so far in the literature, which we denote the *traditional* online problems. In other words, we can vary the "online-ness" of the problems with these disclosure dates.

By introducing disclosure dates, we have defined a new optimization problem: the "online TSP with disclosure dates". We measure the quality of algorithms for this problem using the competitive ratio; the denominator of this ratio is again the optimal value of the TSP with release dates since disclosure dates are irrelevant in the offline situation. For a variety of disclosure date scenarios, we give new online algorithms and derive improved competitive ratios (with respect to the ratios for the traditional online problems), which are functions of the advanced information. In this way, we quantify the *value* of the advanced information given by the disclosure dates. We almost exclusively consider the case where there is a "fixed amount" of advanced notice for each city. In this case, we introduce α and β, two convenient problem parameters that relate the advanced information to the "dimensions" (time and space) of the traditional online problems (exact definitions of α and β will be

given later); we quantify the value of the advanced information in terms of these two parameters. We first detail our results for the online TSP. When cities are contained on the non-negative real line, we give an algorithm that is $\max\{1, \frac{3}{2} - \alpha\}$-competitive and we also prove that this result is best-possible for our disclosure date structure. These results improve upon the $\frac{3}{2}$-competitiveness of a best-possible online algorithm in the traditional case. For the general situation, where cities belong to an arbitrary metric space, we give an algorithm that is $(2 - \frac{\alpha}{1+\alpha})$-competitive. This result improves upon the 2-competitiveness of a best-possible online algorithm for the traditional metric case. We also show a similar result for the multi-salesman online TSP. Next, we consider the online TRP. We analyze a deterministic algorithm and show it is $((1+\sqrt{2})^2 - \frac{\alpha\beta}{\alpha+\beta})$-competitive, where $(1+\sqrt{2})^2$ is the best provable worst-case ratio to-date for the traditional online problem (though this latter result is probably not best-possible). We also give a very similar result for a randomized modification of the previous algorithm; we show this variant is $(\Theta - \frac{\alpha\beta}{\alpha+\beta})$-competitive, where Θ is the traditional competitiveness result.

We next consider speed and vehicle augmentation for generalizations of the online TSP. For single server problems (with both precedence and capacity constraints) with speed augmentation, we give a polynomial-time online algorithm with a competitive ratio of at most $1 + (2\rho - 1)/\gamma$, where $\gamma \geq 1$ is the speed of the online server (the offline server moves at unit speed). For multiple server problems we consider speed and server augmentation (without precedence and capacity constraints); we show that the competitive ratio is at most $1 + \rho(1 - (m-1)\phi)/\gamma + (\rho - 1)/\gamma$, where m is the number of online servers (the offline has a single server) and ϕ is a measure of the problem data. If both online and offline algorithms have m servers at their disposal, our algorithm is again $1 + (2\rho - 1)/\gamma$-competitive.

We then consider constraint augmentation for the online k-TSP. Recall that the k-TSP problem consists of finding a subset of cities of size k to visit. The constraint augmentation is as follows: the online algorithm is only required to visit a subset of size $k' < k$ where the offline must visit k cities. We also consider speed augmentation for the online k-TSP. We again quantify the improvement in the competitive ratio.

Finally, let us mention that we are not certain our improved competitive ratios are best possible. For the online TSP with disclosure dates, we are able to prove a lower bound on the competitive ratio of $(2-\frac{2\alpha}{1+\alpha})$. For the online TRP with disclosure dates, we have no lower bounds. We do not have lower bounds for speed, vehicle and constraint resource augmentation either.

Research in this chapter is joint work with Patrick Jaillet.

1.4.4 Chapter 5: Stochastic Asymptotic Analysis

In this chapter, we take a new approach to studying online routing optimization problems. We introduce stochastic structures to the problem data that are unknown and unused by the online algorithm. We study the "almost sure" properties of a number of online algorithms for variants of the online TSP and TRP as well as certain online machine scheduling problems; for the online TSP problems, we allow for precedence and capacity constraints (but not both) and for the online TRP problems we allow for precedence constraints.

We first consider capacity and speed augmentation (with no precedence constraints) when city locations are stochastic (and release dates arbitrary); we show that our polynomial-time online algorithm has an asymptotic competitive ratio of at most $1 + (\rho q)/(\gamma Q) + (\rho - 1)/\gamma$, almost surely, where Q (q) is the capacity of the online (offline) server.

The remaining results of this chapter consider the case when both city locations and release dates are stochastic. These results are of a similar flavor, which we now describe. We are given an online optimization problem and an algorithm to solve it online. The problem data is stochastic, though unknown to the online algorithm. Let the random variables Z_n^A denote the cost of online algorithm A and Z_n^* the optimal offline cost, both for a problem that is intrinsically of size n (e.g., number of cities for the online TSP).

Trivially, we know that $Z_n^A \geq Z_n^*$. However, we show that, for a number of online routing and scheduling problems under mild probabilistic restrictions, there exist

algorithms A such that

$$\lim_{n\to\infty} \frac{Z_n^A}{Z_n^*} = 1 \text{ a.s.} \tag{1.1}$$

In other words, we prove the almost sure asymptotic optimality of algorithms for these online optimization problems. Our analysis may also be interpreted in a different way. Let the random variable C_n^A denote the cost of being online for algorithm A; i.e., $Z_n^A = Z_n^* + C_n^A$. Next, consider the following restatement of Equation (1.1):

$$\lim_{n\to\infty} \frac{C_n^A}{Z_n^*} = 0 \text{ a.s.}$$

Intuitively, we show that the cost of being online grows at a rate (with respect to problem size) that is strictly smaller than that of the optimal offline cost; i.e., $C_n^A = o(Z_n^*)$ a.s.

We show results of this type for a number of significantly different online routing problems under a variety of stochastic assumptions. We also give the first asymptotic optimality results for the online machine scheduling problems $1|r_j, pmtn|\sum_j w_j C_j$, $1|r_j|\sum_j w_j C_j$, $Q|r_j, pmtn|\sum_j C_j$ and $P|r_j|\sum_j C_j$.

We also characterize the speed of convergence. To better explain our approach, we first give a simple example demonstrating our general technique. Let $X_1, \ldots, X_n$ be i.i.d. random variables of zero mean and unit variance and let $S_n = X_1 + \cdots + X_n$. By the Strong Law of Large Numbers, we know that $S_n/n \to 0$ almost surely. However, if we multiply S_n/n by $\sqrt{n}$, by the Central Limit Theorem, we know that $\sqrt{n}S_n/n = S_n/\sqrt{n} \to N(0,1)$ in distribution. We then say that the convergence of S_n/n to 0 is of the order $1/\sqrt{n}$ in distribution (the weaker convergence of the two results).

We take a similar approach to characterize the rates of convergence of the online algorithms that we study. We know that the ratio ρ_n of the online algorithm's cost to that of the optimal offline algorithm satisfies $(\rho_n - 1) \to 0$, almost surely. We then find the smallest value α, depending on the stochastic model, distribution and algorithm utilized, such that $n^\alpha(\rho_n - 1) \to Z$, where Z is either a non-zero constant or non-degenerate random variable. We then say that the order of convergence is $n^{-\alpha}$, either almost surely or in distribution.

Research in this chapter is joint work with Patrick Jaillet.

1.4.5 Chapter 6: Computational Results

In this chapter, we perform a computational study of the online TSP, TRP and certain machine scheduling problems. We show how the online algorithms perform on a number of different problem instances. Additionally, we exhibit concretely the asymptotic optimality derived theoretically in Chapter 5. Interestingly, these computational results also suggest that, in practice, the quality of the output of our algorithms will be much better than what the theoretical bounds predict.

Research in this chapter is joint work with Patrick Jaillet and José Correa.

1.4.6 Chapter 7: Conclusion and Future Work

In this chapter we first summarize the thesis and then we give a number of directions for further research.

Chapter 2

Online Routing Optimization Problems

2.1 Introduction

We begin this chapter by recalling the definition of the TSP: We are given a metric space and a set of points in the space, representing cities. Given an origin city, the task is to find a tour of minimum total length, beginning and ending at the origin, that traverses each city at least once. Assuming a constant speed, we can interpret this objective as minimizing the *time* required to complete a tour. We may also incorporate release dates, where a city must be visited on or after its release date; in this case the problem is known as the "TSP with release dates." The problem becomes online when cities are only revealed at their release dates.

Additional constraints can be added to the above salesman problems. We can introduce precedence constraints where some cities must be visited before others. The salesman can be considered a vehicle/server that transports packages. In this case it is natural to introduce a capacity for the server; in other words, a server can visit only a subset of all cities in a given tour and must traverse multiple tours in order to serve all requests. Finally, we also consider the case where we have multiple servers to control.

In this chapter we are concerned with online versions of the above mentioned

routing optimization problems. In our framework, the problem data is revealed dynamically over time, independent of the server's location, at release dates. The assumption that problem instances are completely known a priori is unrealistic in many applications. Taxi services, buses and courier services, for example, require an online model in which locations to be visited are revealed over time, while the server is en route serving previously released requests. The focus of this chapter is on studying algorithms for a variety of online routing problems. These algorithms are evaluated using competitive analysis and, in particular, the competitive ratio, which is defined as the worst case ratio of the online algorithm's cost to the cost of an optimal offline algorithm, where all data is known a priori.

We design online algorithms for new online routing problems and we derive new competitive ratio bounds. A number of our competitive ratio results are the first for certain online routing problems. We also study the competitive ratios of polynomial-time online algorithms; in all cases we are able to relate the competitive ratio to the approximation ratio of a simpler problem.

Outline: In Section 2.2 we present an algorithm from the literature for the online TSP and we generalize it so that it can run in polynomial time. In Section 2.3 we generalize the online TSP, allowing for precedence and capacity constraints; we design and analyze algorithms for this new problem. In Section 2.4, we generalize the online TSP by allowing for multiple salesmen; again, we design and analyze algorithms for this new problem. Finally, in Sections 2.5 and 2.6, we present results from the literature for the online Traveling Repairman Problem (TRP) as well as the online k-TSP; these results are presented here for comparison because we later generalize them in our study of resource augmentation in Chapter 4.

2.2 The Online TSP and Algorithm PAH

Let us first state the assumptions and definitions we utilize in this section.

1. City locations belong to some metric space $\mathcal{M}$.

2. A city is revealed to the salesman at its release date.

3. A city is ready for service at its release date. The service requirement at a city is zero.

4. The salesman travels at unit speed or is idle.

5. The problem begins at time 0, and the salesman is initially at a designated origin o of the metric space.

6. The online TSP objective is to minimize the time required to visit all cities and return to the origin.

The problem data is a set of points (l_i, r_i), $i = 1, \ldots, n$, where n is the number of cities. The quantity $l_i \in \mathcal{M}$ is the i^{th} city's location. The quantity $r_i \in \mathbb{R}_+$ is the i^{th} city's release date; i.e., r_i is the first time after which that city i will accept service. We also let $\mathcal{N} = \{1, \ldots, n\}$. We index the cities such that $r_1 \leq r_2 \leq \cdots \leq r_n$.

From the online perspective, the total number of requests, represented by the parameter n, is not known, and city i only becomes known at time r_i. Z_n^A will denote the cost of online algorithm A on an instance of n cities and Z_n^* is the optimal offline cost on n cities (at times, the n term will be suppressed). Finally, define L_{TSP} as the optimal TSP (without release dates) tour length through all cities in an instance.

We now present the online algorithm Plan-At-Home (PAH), which was given in [10], as well as a relevant theorem.

Algorithm 1 : PAH

(1) Whenever the salesman is at the origin, it starts to follow an optimal route that serves all the requests yet to be served and goes back to the origin.

(2) If at time r_i, for some i, a new request is presented at point l_i, then it takes one of two actions depending on its current position p:

(2a) If $d(l_i, o) > d(p, o)$, then the salesman goes back to the origin where it appears in a Case 1 situation.

(2b) If $d(l_i, o) \leq d(p, o)$, then the salesman ignores it until it arrives at the origin, where again it re-enters Case 1.

Theorem 3 ([10]) *The competitive ratio of PAH is 2. Furthermore, there does not exist another online algorithm with a strictly smaller competitive ratio.*

2.2.1 Polynomial-time Online Algorithms: PAH-p

We now give our first generalization of algorithm PAH ([10]), which we shall denote PAH-p as all subroutines are polynomial-time.

Algorithm 2 : PAH-p

(1) Whenever the salesman is at the origin, it starts to follow a tour that serves all cities whose release dates have passed but have not yet been served; this tour is constructed using a ρ-approximation algorithm A that solves an offline TSP (with no release dates).

(2) If at time r_i, for some i, a new city is presented at point l_i, the salesman takes one of two actions depending on the salesman's current position p:

(2a) If $d(l_i, o) > d(p, o)$, the salesman goes back to the origin where it appears in a Case (1) situation.

(2b) If $d(l_i, o) \leq d(p, o)$, the salesman ignores the city until it arrives at the origin, where again it re-enters Case (1).

Theorem 4 *Algorithm PAH-p is 2ρ-competitive.*

Proof Recall that r_n is the time of the last request, l_n the position of this request and $p(t)$ the location of the salesman at time t. We show that in each of the Cases (1), (2a) and (2b), PAH-p is 2ρ-competitive.

Case (1): PAH-p is at the origin at time r_n. Then it starts a ρ-approximate tour that serves all the unserved requests. The time needed by PAH-p is at most $r_n + \rho L_{TSP} \leq (1 + \rho)Z^* \leq 2\rho Z^*$.

Case (2a): We have that $d(o, l_n) > d(o, p(r_n))$. PAH-p returns to the origin, where it will arrive before time $r_n + d(o, l_n)$. After this, PAH-p computes and follows a ρ-approximate tour through all the unserved requests. Therefore, the online cost is at most $r_n + d(o, p(r_n)) + \rho L_{TSP} < r_n + d(o, l_n) + \rho L_{TSP}$. Noticing that $r_n + d(o, l_n) \leq Z^*$, we have that the online cost is at most $(1 + \rho)Z^* \leq 2\rho Z^*$.

Case (2b): We have that $d(o, l_n) \leq d(o, p(r_n))$. Suppose PAH-p is following a route $\mathcal{R}$ that had been computed the last time it was at the origin. Note that $\mathcal{R} \leq \rho L_{TSP} \leq \rho Z^*$. Let $\mathcal{Q}$ be the set of requests temporarily ignored since the last time PAH-p was at the origin; $\mathcal{Q}$ is not empty since it contains l_n. Let l_q be the location of the first request in $\mathcal{Q}$ served by the offline algorithm and let r_q be the time at which l_q was released. Let $\mathcal{P}_\mathcal{Q}$ be the shortest path that starts at l_q, visits all the cities in $\mathcal{Q}$ and ends at o. Clearly, $Z^* \geq r_q + \mathcal{P}_\mathcal{Q}$ and $Z^* \geq d(o, l_q) + \mathcal{P}_\mathcal{Q}$.

At time r_q, the distance that PAH-p still has to travel on the route $\mathcal{R}$ before arriving at o is at most $\mathcal{R} - d(o, l_q)$, since $d(o, p(r_q)) \geq d(o, l_q)$ implies that PAH-p has traveled on the route $\mathcal{R}$ a distance not less than $d(o, l_q)$. Therefore, it will arrive at o before time $r_q + \mathcal{R} - d(o, l_q)$. After that it will follow a ρ-approximate tour $\mathcal{T}_\mathcal{Q}$ that covers the set $\mathcal{Q}$; letting $\mathcal{T}^*_\mathcal{Q}$ be the optimal tour over the set $\mathcal{Q}$, we have that $\mathcal{T}_\mathcal{Q} \leq \rho \mathcal{T}^*_\mathcal{Q}$. Hence, the completion time will be at most $r_q + \mathcal{R} - d(o, l_q) + \rho \mathcal{T}^*_\mathcal{Q}$. Since $\mathcal{T}^*_\mathcal{Q} \leq d(0, l_q) + \mathcal{P}_\mathcal{Q}$, we have that the online cost is at most

$$\begin{aligned} r_q + \mathcal{R} - d(o, l_q) + \rho d(0, l_q) + \rho \mathcal{P}_\mathcal{Q} &= (r_q + \mathcal{P}_\mathcal{Q}) + \mathcal{R} + (\rho - 1)(d(0, l_q) + \mathcal{P}_\mathcal{Q}) \\ &\leq Z^* + \rho Z^* + (\rho - 1)Z^* \\ &= 2\rho Z^*. \blacksquare \end{aligned}$$

Applying well-known algorithms given in [25] and [6], we are able to give two interesting corollaries.

Corollary 1 *If we choose A as Christofides' heuristic, Algorithm PAH-p is 3-competitive.*

This matches the 3-competitive polynomial-time algorithm given in [10]. However, a polynomial-time algorithm with a competitive ratio of at most 2.78 was recently

given in [9].

Corollary 2 *If $\mathcal{M}$ is Euclidean and we choose A as Arora's PTAS, for any $\epsilon > 0$, Algorithm PAH-p is $(2+\epsilon)$-competitive.*

To the best of our knowledge, this is the first result for the online TSP in the Euclidean metric space.

2.3 Generalized Online Routing I: Precedence and Capacity Constraints and Algorithm PAH-G

We now consider routing problems where a single server must service a more complex sequence of requests. The data for our problems is a set of points $(\mathbf{l}_i, r_i, \mathbf{d}_i)$, $i = 1, \ldots, n$, where n is the number of requests and $k(i)$ is the number of cities in request i: $\mathbf{l}_i = (l_i^1, l_i^2, \ldots, l_i^{k(i)})$ and $\mathbf{d}_i = (d_i^1, d_i^2, \ldots, d_i^{k(i)})$. The quantity $l_i^j \in \mathcal{M}$, $\mathcal{M}$ an arbitrary metric space, is the location of the j^{th} city in the i^{th} request. The quantity $r_i \in \mathbb{R}_+$ is the i^{th} request's release date; i.e., r_i is the first time after which cities in request i will accept service. The quantity $d_i^j \in \mathbb{R}_+$ is the demand[1] of city l_i^j. The server has a capacity Q; the sum of city demands visited on any given tour can be at most Q; we assume $d_i^j \leq Q$ for all i, j. Precedence constraints exist *within* a request; i.e., for a fixed i, arbitrary precedence constraints of the form $l_i^j \preceq l_i^k$ (l_i^j must be visited before l_i^k) can exist, for any $j \neq k$. The service requirement at a city is zero. The server travels at unit speed or is idle. The problem begins at time 0, and the server is initially at a designated origin o of the metric space. The objective is to minimize the time required to visit all cities and have the server return to the origin.

From the online perspective, the total number of requests, represented by the parameter n, is not known, and request i only becomes known at time r_i. $Z_n^A(Q)$ denotes the cost of online algorithm A on an instance of n cities with server capacity Q and $Z_n^*(Q)$ is the corresponding optimal offline cost where all data is known a

[1]It is possible to generalize our capacity model to allow positive and negative demands as well as different types of products being transfered. However, we study the current problem to limit the complexity of the analysis.

priori. $Z_n^{r=0}(Q)$ is the optimal cost when all release dates are equal to zero; clearly, $Z_n^{r=0}(Q) \leq Z_n^*(Q)$. The problem instance underlying $Z_n^{r=0}(Q)$, $Z_n^A(Q)$ and $Z_n^*(Q)$ will be clear from context. At times, the n term will be suppressed. Recall that L_{TSP} is defined as the optimal TSP tour length through all cities in an instance; i.e., $L_{TSP} = Z^{r=0}(\infty)$.

We give an online algorithm that generalizes PAH ([10]); we denote our algorithm Plan-At-Home-Generalized (PAH-G). Recall that the competitive ratio of PAH is 2.

Algorithm 3 : PAH-G

(1) Whenever the server is at the origin, it calculates and implements a ρ-approximate solution to $Z^{r=0}(Q)$ over all requests whose release dates have passed but have not yet been served completely.

(2) If at time r_i, for some i, a new request is presented, the server takes one of two actions depending on the server's current position p and the farthest location in the current request l_i^*:

$$l_i^* = \arg \max_{\{l_i^j \mid 1 \leq j \leq k(i)\}} d(o, l_i^j) :$$

(2a) If $d(l_i^*, o) > d(p, o)$, the server goes back to the origin where it appears in a Case (1) situation.

(2b) If $d(l_i^*, o) \leq d(p, o)$, the server ignores request i until it completes the route it is currently traversing, where again Case (1) is encountered.

Theorem 5 *Algorithm PAH-G is 2ρ-competitive.*

Proof Recall that r_n is the time of the last request and $l_n^* = \arg\max_{\{l_n^j \mid 1 \leq j \leq k(n)\}} d(o, l_n^j)$. We show that in each of the Cases (1), (2a) and (2b), PAH-G is 2ρ-competitive.

In Case (1) PAH-G is at the origin at time r_n. It starts traversing a ρ-approximate set of tours that serve all the unserved requests. The time needed by PAH-G is at most $r_n + \rho Z_n^{r=0}(Q) \leq (1 + \rho) Z_n^*(Q)$.

Considering Case (2a), we have that $d(o, l_n^*) > d(o, p)$. Then PAH-G goes back to the origin, where it will arrive before time $r_n + d(o, l_n^*)$. After this, PAH-G computes and follows a ρ-approximate set of tours through all the unserved requests. Therefore, the online cost is at most $r_n + d(o, l_n^*) + \rho Z_n^{r=0}(Q)$. Noticing that $r_n + d(o, l_n^*) \leq Z_n^*(Q)$, we have that the online cost is at most $(1 + \rho)\, Z_n^*(Q)$.

Finally, we consider Case (2b), where $d(o, l_n^*) \leq d(o, p)$. Suppose PAH-G is following a route $\mathcal{R}$ that had been computed the last time step (1) of PAH-G had been invoked. $\mathcal{R}$ will also denote the actual distance of the route; we have that $\mathcal{R} \leq \rho Z^{r=0}(Q) \leq \rho Z^*(Q)$. Let $\mathcal{S}$ be the set of requests that have been temporarily ignored (from step (2b)) since the last time PAH-G invoked step (1). Let l_f be the first location of the first request in $\mathcal{S}$ visited by the offline algorithm, and let r_f be the time at which request f was released. Let $\mathcal{P}_{\mathcal{S}}^*$ be the fastest route that starts at l_f, visits all cities in $\mathcal{S}$ and ends at the origin, respecting precedence and capacity constraints. Clearly, $Z^*(Q) \geq r_f + \mathcal{P}_{\mathcal{S}}^*$ and $Z^*(Q) \geq d(o, l_f) + \mathcal{P}_{\mathcal{S}}^*$.

At time r_f, the time that PAH-G still has left to complete route $\mathcal{R}$ is at most $\mathcal{R} - d(o, l_f)$, since $d(o, p(r_f)) \geq d(o, l_f^*) \geq d(o, l_f)$ implies that PAH-G has traveled on the route $\mathcal{R}$ a distance not less than $d(o, l_f)$. Therefore, the server will complete the route $\mathcal{R}$ before time $r_f + \mathcal{R} - d(o, l_f)$. After that it will follow a ρ-approximate set of tours that covers the set $\mathcal{S}$ of yet unserved requests; let $\mathcal{T}_{\mathcal{S}}$ denote the cost of the *optimal* set of tours. Hence, the total time to completion will be at most $r_f + \mathcal{R} - d(o, l_f) + \rho \mathcal{T}_{\mathcal{S}}$. Since $\mathcal{T}_{\mathcal{S}} \leq d(0, l_f) + \mathcal{P}_{\mathcal{S}}^*$, we have that the online cost is at most

$$\begin{aligned} r_f + \mathcal{R} - d(o, l_f) + \rho d(0, l_f) + \rho \mathcal{P}_{\mathcal{S}}^* &= (r_f + \mathcal{P}_{\mathcal{S}}^*) + \mathcal{R} + (\rho - 1)\, d(0, l_f) + (\rho - 1)\, \mathcal{P}_{\mathcal{S}}^* \\ &\leq Z^*(Q) + \rho Z^{r=0}(Q) + (\rho - 1)\, Z^*(Q) \\ &\leq 2\rho Z^*(Q). \blacksquare \end{aligned}$$

As an example, if we consider the online capacitated TSP without precedence constraints, we can apply the Iterated Tour Partition (ITP) heuristics given by Altinkemer and Gavish [3] and Haimovich and Rinnooy Kan [38]. If $d_i^j = 1$ for all

i, j, there exists a ITP heuristic with approximation ratio $\rho \leq (5/2 + 3/2Q)$. If demands are arbitrary integers, there exists a ITP heuristic with approximation ratio $\rho \leq (7/2 - 3/Q)$.

Corollary 3 *If we use an exact algorithm in step(1) for calculating an optimal offline* $Z^{r=0}(Q)$*, Algorithm PAH-G is 2-competitive.*

These results show interesting properties. First, it is possible to relate the competitive ratio to the approximation ratio of a simpler but related optimization problem $Z^{r=0}(Q)$. Also, if we have access to exact algorithms for $Z^{r=0}(Q)$, adding capacity and precedence constraints results in no increase in the competitive ratio, with respect to the online TSP.

2.4 Generalized Online Routing II: Multiple Vehicle Routing and Algorithm PAH-m

We next consider routing problems with m identical servers. We do not consider capacity or precedence constraints. The data for our multiple server problems is closely related to that of the single server problems: the data is a set of points (l_i, r_i), $i = 1, \ldots, n$ where $l_i \in \mathcal{M}$ ($r_i \in \mathbb{R}^+$) is the location (release date) of the i-th request. The service requirement at a city is again zero. The servers travel at unit speed or are idle. The problem begins at time 0, and all servers are initially at a designated origin o of the metric space. The objective is to minimize the time required to visit all cities and have all servers return to the origin. Note that each city only needs to be visited by a single server.

$Z_n^A(m)$ denotes the cost of online algorithm A on an instance of n cities with m identical servers and $Z_n^*(m)$ is the corresponding optimal offline cost where all data are known a priori; we assume $n \geq m$. $Z_n^{r=0}(m)$ is the optimal cost when all release dates are equal to zero; clearly, $Z_n^{r=0}(m) \leq Z_n^*(m)$. Note that $Z_n^{r=0}(m)$ is equivalently the problem of finding a set of m tours, that collectively visit all locations, such that the maximum tour length is minimized; e.g., see Frederickson, Hecht and Kim [30]. The

problem instance underlying $Z_n^{r=0}(m)$, $Z_n^A(m)$ and $Z_n^*(m)$ will be clear from context. Finally, note that $L_{TSP} = Z^{r=0}(1)$. The competitive ratio is defined similarly to the single server case.

We again give an online algorithm that generalizes PAH ([10]); we denote our algorithm Plan-At-Home-m-Servers (PAH-m).

Algorithm 4 : PAH-m

(1) Whenever all servers are at the origin, they calculate and implement a ρ-approximate solution to $Z^{r=0}(m)$ over all requests whose release dates have passed but have not yet been served.

(2) If at time r_i, for some i, a new request is presented, the servers take one of two actions depending on the request's location l_i and the farthest server's current position p^* (ties broken arbitrarily):

$$p^* = \arg \max_{\{p_i \mid 1 \le i \le m\}} d(o, p_i) :$$

(2a) If $d(l_i, o) > d(p^*, o)$, all servers go back to the origin where they appear in a Case (1) situation.

(2b) If $d(l_i, o) \le d(p^*, o)$, all servers **except** *p^* return to the origin; server p^* ignores request i until it completes the route it is currently traversing, where again Case (1) is encountered.*

Theorem 6 *Algorithm PAH-m is 2ρ-competitive.*

Proof Recall that r_n is the time of the last request, l_n the position of this request and $p^*(t)$ the location of the farthest salesman at time t.

Case (1): All salesmen are at the origin at time r_n. Then they start implementing a ρ-approximate solution to $Z^{r=0}(m)$ that serves all the unserved requests. The time needed by PAH-m is at most $r_n + \rho Z^{r=0}(m) \le (1+\rho) Z^*(m)$.

Case (2a): We have that $d(o, l_n) > d(o, p^*(r_n))$. All salesmen return to the origin, where they will all arrive before time $r_n + d(o, l_n)$. After this, PAH-m computes and

follows a ρ-approximate solution to $Z^{r=0}(m)$ through all unserved requests. Therefore, the online cost is at most $r_n + d(o, l_n) + \rho Z^{r=0}(m)$. Noticing that $r_n + d(o, l_n) \leq Z^*(m)$, we have that the online cost is at most $(1 + \rho)\, Z^*(m)$.

Case (2b): We have that $d(o, l_n) \leq d(o, p^*(r_n))$ and all salesmen, except p^*, return to the origin, if not yet already there. Suppose salesman p^* is following a tour $\mathcal{R}$ that had been computed the last time it was at the origin. Note that $\mathcal{R} \leq \rho Z^{r=0}(m)$. Let $\mathcal{Q}$ be the set of requests temporarily ignored since the last time a Case (1) re-optimization was performed; since $l_n \in \mathcal{Q}$, $\mathcal{Q}$ is not empty. Let $\mathcal{S} \subseteq \{1, \ldots, m\}$ denote the set of salesmen that serve $\mathcal{Q}$ in the optimal offline solution. For $j \in \mathcal{S}$, let l^j be the location of the first city in $\mathcal{Q}$ served by server j in the *optimal offline solution* and let r^j be the time at which this city was released. Let $\mathcal{P}^j_{\mathcal{Q}}$, $j \in \mathcal{S}$, be the set of paths, the j-th path starting from l^j, that collectively visit all the cities in $\mathcal{Q}$ and end at the origin, such that the maximum path length is minimized (ties broken arbitrarily). It is easy to see that $Z^*(m) \geq \max_{j \in \mathcal{S}} \{\mathcal{P}^j_{\mathcal{Q}}\}$ since the min-max-path optimization has distinct advantages over the offline solution: (1) having the servers start at cities l^j, (2) needing to only serve the cities in $\mathcal{Q}$ and (3) ignoring release dates. If the servers start from the origin, the earliest time that server j can visit city l^j is $\max\{r^j, d(0, l^j)\}$; by extension we have that $Z^*(m) \geq \max_{j \in \mathcal{S}} \{r^j + \mathcal{P}^j_{\mathcal{Q}}\}$ and $Z^*(m) \geq \max_{j \in \mathcal{S}} \{d(o, l^j) + \mathcal{P}^j_{\mathcal{Q}}\}$.

At time r^j, the distance that salesman p^* still has to travel on the route $\mathcal{R}$ before arriving at the origin is at most $\mathcal{R} - d(o, l^j)$, since $d(o, p^*(r^j)) \geq d(o, l^j)$ implies that p^* has traveled on the route $\mathcal{R}$ a distance not less than $d(o, l^j)$. Therefore, it will arrive at the origin before time $r^j + \mathcal{R} - d(o, l^j)$; note that since this is valid for any j, we can say that the salesman will arrive at the origin before time $\min_{j \in \mathcal{S}} \{r^j + \mathcal{R} - d(o, l^j)\}$. Note that all other salesmen have already arrived at the origin. Next, a ρ-approximate $Z^{r=0}(m)$ will be implemented on $\mathcal{Q}$; let $\mathcal{T}_{\mathcal{Q}}$ denote the *optimal* maximum tour length. Hence, the completion time of PAH-m will be at most $\min_{j \in \mathcal{S}} \{r^j + \mathcal{R} - d(o, l^j)\} + \rho \mathcal{T}_{\mathcal{Q}}$. Now, note the following feasible solution for the final case (1) re-optimization: Use only the set of salesmen $\mathcal{S}$, force salesman j to first go to city l^j and then traverse path $\mathcal{P}^j_{\mathcal{Q}}$. Therefore, $\mathcal{T}_{\mathcal{Q}} \leq \max_{j \in \mathcal{S}} \{d(0, l^j) + \mathcal{P}^j_{\mathcal{Q}}\}$ and we have that the online cost is

at most

$$\min_{j\in\mathcal{S}}\left\{r^j+\mathcal{R}-d(o,l^j)\right\}+\rho\max_{j\in\mathcal{S}}\left\{d(0,l^j)+\mathcal{P}_{\mathcal{Q}}^j\right\}.$$

Letting k be the arg max of the second term, we have that the online cost is at most

$$\begin{aligned}
& r^k+\mathcal{R}-d(o,l^k)+\rho\left(d(0,l^k)+\mathcal{P}_{\mathcal{Q}}^k\right)\\
= \;& \left(r^k+\mathcal{P}_{\mathcal{Q}}^k\right)+\mathcal{R}+(\rho-1)\,d(o,k^k)+(\rho-1)\,\mathcal{P}_{\mathcal{Q}}^k\\
\leq \;& Z^*(m)+\rho Z^{r=0}(m)+(\rho-1)\left(d(o,l^k)+\mathcal{P}_{\mathcal{Q}}^k\right)\\
\leq \;& 2\rho Z^*(m).\ \blacksquare
\end{aligned}$$

As an example, we can apply the approximation algorithm for $Z^{r=0}(m)$ given by Frederickson, Hecht and Kim [30] that has an approximation ratio $\rho \leq 5/2 - 1/m$.

Corollary 4 *If we use an exact algorithm in step(1) for calculating an optimal offline $Z^{r=0}(m)$, the competitive ratio of PAH-m is 2.*

Proof By choosing $\rho = 1$ in Theorem 6, an upper bound of 2 on the competitive ratio is clear. We next provide a lower bound.

Define a metric space $\mathcal{M}$ as a graph with vertex set $V = \{1, 2, \ldots, n\} \cup \{o\}$ with distance function d that satisfies the following: $d(o,i) = 1$ and $d(i,j) = 2$ for all $i \neq j \in V \setminus \{o\}$. For simplicity, assume m divides n evenly.

At time 0, there is a request at each of the n cities in $V \setminus \{o\}$. If an online server visits the request at city i at time $t \leq 2\frac{n}{m} - 1 - \epsilon$, for some small ϵ, then at time $t+\epsilon$, a new request is released at city i. In this way, at time $2\frac{n}{m}-1$ the online servers still have to serve requests at all n cities and will finish at time $(2\frac{n}{m}-1)+1+2(\frac{n}{m}-1) = 4\frac{n}{m}-2$. Therefore, denoting C_A as the online cost of an arbitrary online algorithm A, we have that $C_A \geq 4\frac{n}{m} - 2$. The optimal offline servers, however, will be able to visit all cities by time $2\frac{n}{m}$. Therefore, by letting $k = \frac{n}{m}$, we have that $\frac{C_A}{C_{\mathrm{OPT}}} \geq \frac{4k-2}{2k}$; taking k arbitrarily large proves the theorem. $\blacksquare$

Again, it is possible to relate the competitive ratio to the approximation ratio of a simpler but related optimization problem $Z^{r=0}(m)$. Also, if we have access to

exact offline algorithms for $Z^{r=0}(m)$, adding extra salesmen to the problem definition results in no change (increase or decrease) in the competitive ratio, with respect to the the online TSP.

2.5 The Online TRP

Thus far, we have been analyzing versions of the online TSP, where the objective is arguably in the salesman's interest. We now consider another objective, the weighted latency, which is an objective that is arguably in the cities' interest; additionally, we are able to favor certain cities over others.

In this section, we consider the online Traveling Repairman Problem (TRP) with arbitrary weights. We first generalize our data: the problem data is now (l_i, r_i, w_i), $i = 1, \ldots, n$ where $w_i \in \mathbb{R}^+$ are the weights of the cities. A city's weight is revealed along with its locations at the city's release date.

Our objective is to minimize $\sum_{i \in \mathcal{N}} w_i C_i$, where C_i is the completion time of city i, the first time it is visited after its release date. Again, $l_i \in \mathcal{M}$, for any metric space $\mathcal{M}$.

We now present the algorithm INTERVAL given in [51] as well as a relevant theorem. Note that algorithm INTERVAL is also valid for the online dial-a-ride problem, a generalization of the online TRP.

Algorithm 5 : INTERVAL *Phase 0: In this phase the algorithm is initialized.*

Set L to be the earliest time when a request could be completed by an optimal offline algorithm. We can assume that $L > 0$, since $L = 0$ means that there are requests released at time 0 with source and destination o. These requests are served at no cost. Until time L, remain at the origin o. For $i = 0, 1, 2, \ldots$, define $B_i = \lambda^{i-1} L$.

Phase i, for $i = 1, 2, \ldots$: At time B_i compute a transportation schedule S_i for the set of yet unserved requests released up to time B_i with the following properties:

1. *Schedule S_i starts at the endpoint x_{i-1} of schedule S_{i-1} (we set $x_0 = o$).*

2. *Schedule S_i ends at a point x_i with an empty server such that $d(o, x_i) \leq B_i$.*

3. *The length of schedule S_i, denoted by $l(S_i)$, satisfies*

$$l(S_i) \leq \begin{cases} B_1, & i = 1 \\ B_i + B_{i-1}, & i \geq 2 \end{cases}$$

4. *The transportation schedule S_i maximizes the sum of the weights of requests served among all schedules satisfying 1-3.*

If $i = 1$, then follow S_1 starting at time B_1. If $i \geq 2$, follow S_i starting at time βB_i until time βB_{i+1}, where $\beta = (\lambda + 1)/(\lambda(\lambda - 1))$.

Theorem 7 ([51]) *Algorithm INTERVAL is $\lambda(\lambda + 1)/(\lambda - 1)$-competitive for any $\lambda \in (1, 1 + \sqrt{2}]$. With $\lambda = (1 + \sqrt{2})$, algorithm INTERVAL is $(1 + \sqrt{2})^2$-competitive*[2].

[51] also analyzed a randomized version of the above algorithm, which they denote RANDINTERVAL. The only difference is in the initialization of the algorithm: $L \mapsto \lambda^U L$, where U is random variable uniformly distributed between 0 and 1.

Theorem 8 ([51]) *Algorithm RANDINTERVAL is $(\lambda+1)/\ln\lambda$-competitive for any $\lambda \in (1, 1 + \sqrt{2}]$. With $\lambda = (1 + \sqrt{2})$, algorithm RANDINTERVAL is $(2 + \sqrt{2})/\ln(1 + \sqrt{2})$-competitive* [3].

2.6 The Online k-TSP

In this final section we return to a variant of the online TSP, the online k-TSP. The problem data is again (l_i, r_i), $i = 1, \ldots, n$. We are also given an integer $k \in \mathbb{Z}^+$ and the objective is to choose a subset of cities of size k and minimize the time needed to visit the cities and return to the origin. We present the algorithm Wait-and-Go (WaG) given in [9]. Note that their algorithm is also valid for the online Quota-TSP,

[2]The range for λ in this theorem was originally given in [51] as $[1 + \sqrt{2}, 3]$, which is incorrect since β is required to be greater than or equal to one. This correction was given in [43]; see also [52] and Chapter 4 of this thesis.

[3]The original result in [51], based on an incorrect range for λ (see other footnote on this page), was picking $\lambda = 3$ as the best λ, which resulted in $4/\ln 3$-competitiveness where $4/\ln 3 < (2 + \sqrt{2})/\ln(1 + \sqrt{2})$. This correction was given in [43]; see also [52] and Chapter 4 of this thesis.

where each city has a demand and the objective is to cover a given quota of city demands.

Algorithm 6 : WaG *For any (partial) sequence σ of at least k cities already presented to the server, compute the optimal value $Z^*(\sigma)$. At time $t = Z^*(\sigma)$ the algorithm implements $Z^*(\sigma)$.*

Theorem 9 ([9]) *The competitive ratio of algorithm WaG, in general metric spaces, is 2. Furthermore, there does not exist another online algorithm with strictly smaller competitive ratio.*

Chapter 3

Online Machine Scheduling Problems

3.1 Introduction

In this chapter, we study online versions of classic parallel machine scheduling problems. Given a set of jobs $N = \{1, \ldots, n\}$, where each job j has a processing time $p_j > 0$, a weight $w_j > 0$ and a release date $r_j \geq 0$, we want to process these jobs on m identical machines. We consider both non-preemptive and preemptive versions; in the latter case, a job being processed may be interrupted and resumed later, possibly on a different machine. Letting C_j be the completion time of job j in a given schedule, we are interested in minimizing the weighted sum of completion times: $\sum_{j \in N} w_j C_j$. As we only consider online problems, jobs are not known until their respective release date. In scheduling notation Graham, Lawler, Lenstra and Rinnooy Kan [36], we consider online versions of $P|r_j| \sum_j w_j C_j$ and $P|r_j, pmtn| \sum_j w_j C_j$.

In online optimization we are dealing with limitations on information, contrasting with the limitations on computational power in classic approximation algorithm design. The standard measures of quality of online algorithms is the competitive ratio and the notion of competitiveness. Similarly to the approximation guarantee of an algorithm, the competitive ratio is defined to be the worst case ratio, over all instances, of the cost output by the online algorithm to the optimal offline cost;

for more details, see Chapter 1, Section 1.2. In some situations randomization is a powerful tool to obtain algorithms with better performance ratios. The competitive ratio of a randomized online algorithm (against an oblivious adversary; see Chapter 1, Section 1.2) is defined as above replacing "the cost output by the online algorithm" by the *expected* cost output by the online algorithm.

Outline: We begin in Section 3.2 by giving notation and existing results that are utilized throughout this chapter. In Section 3.3 we design and analyze a deterministic online algorithm for $P|r_j|\sum_j w_j C_j$. In Section 3.4, we analyze a randomized variant of the previous algorithm for the same problem. Finally, in Section 3.5 we design and analyze a randomized online algorithm for $P|r_j, pmtn|\sum_j w_j C_j$.

3.2 Preliminaries

According to [58] the α-point $t_j(\alpha)$, $0 < \alpha \leq 1$, of job j in a given schedule is defined as the first time an α-fraction of job j has been completed (i.e., the first time αp_j has been processed). The general idea of our subsequent algorithms is to schedule jobs on the m machines by list-scheduling the jobs in the order of their α-points on a virtual machine, which is "m-times faster". Additionally, these algorithms may use job-dependent α's to guide the schedule; in this latter case, we shall denote job j's alpha value as α_j. The concept of a single fast virtual machine was apparently first considered by Eastman, Even and Isaacs [27]. Recently, [22] considered a "preemptive one-machine relaxation" where jobs are list-scheduled on parallel machines in order of their completion times on a single virtual machine.

Another important ingredient in what follows is related to mean-busy-time relaxations of $1|r_j|\sum w_j C_j$. The *mean busy time* M_j of job j, is defined as the average point in time at which job j is being processed; see Goemans [34] and Goemans [35]. Alternatively it can be computed as $M_j = \int_0^1 t_j(\alpha)d\alpha$. Let $p(S) = \sum_{j\in S} p_j$, $w(S) = \sum_{j\in S} w_j$ and $r_{\min}(S) = \min_{j\in S}\{r_j\}$. Following [35] for a scheduling instance $I = \{(p_i, r_i, w_i),\ i \in N\}$ we define $Z_R(I)$ to be the value of mean busy time relaxation

for $1|r_j, pmtn|\sum_j w_j C_j$; i.e.:

$$\begin{aligned} Z_R(I) &\triangleq \min \sum_{j\in N} w_j M_j \\ \text{subject to} \quad & \sum_{j\in S} p_j M_j \geq p(S)\left(r_{\min}(S) + \frac{1}{2}p(S)\right),\ S \subseteq N. \end{aligned}$$

It was shown in [34] that $Z_R(I)$ can be obtained online by scheduling, at any point in time, the available job j with the highest ratio w_j/p_j. This schedule is called the *LP schedule.*

Now, for an instance $I = \{(p_i, r_i, w_i),\ i \in N\}$ of $P|r_j, pmtn|\sum w_j C_j$ with m parallel machines, let $Z^m(I)$ be the value of the optimal schedule. Consider the instance $I_m = \{(\frac{p_i}{m}, r_i, w_i),\ i \in N\}$ and let $Z_R^m(I) = Z_R(I_m)$ i.e., the value of the mean-busy-time relaxation on I_m (note that this is equivalent to the value of the mean-busy-time relaxation on instance I in a machine that is m times faster). Thus, $Z_R^m(I)$ can be evaluated as:

$$\begin{aligned} Z_R^m(I) &= \min \sum_{j\in N} w_j M_j \\ \text{subject to} \quad & \sum_{j\in S} p_j M_j \geq p(S)\left(r_{\min}(S) + \frac{1}{2m}p(S)\right),\ S \subseteq N. \end{aligned}$$

The following lemma provides a simple, yet powerful, lower bound for $P|r_j, pmtn|\sum w_j C_j$. It is a particular case of a bound obtained in [24] in a more general framework. It was also obtained in [64], expressed in terms of an equivalent time-indexed relaxation.

Lemma 1 ([24],[64]) *For any scheduling instance I, $Z_R^m(I) + \frac{1}{2}\sum_{j\in N} w_j p_j \leq Z^m(I)$.*

To finish this section let us review the concept of canonical decomposition [33] and a useful formula to rewrite $\sum w_j M_j$ [34] (see also [35]). For a set of jobs S, consider a single machine schedule that processes jobs in S as early as possible. This induces a partition of jobs in S into $S_1, \ldots, S_k$ such that the machine is busy exactly in the disjoint intervals $[r_{\min}(S_l), r_{\min}(S_l) + p(S_l)]$, for $l = 1, \ldots, k$. This

partition is the *canonical decomposition* of S. Also, a set is canonical if it equals its canonical decomposition. Assume that $w_1/p_1 \geq \cdots \geq w_n/p_n \geq w_{n+1}/p_{n+1} = 0$ and let $[i] = \{1, \ldots, i\}$. Consider $S_1^i, \ldots, S_{k(i)}^i$, the canonical decomposition of $[i]$; then for any vector $M = (M_1, \ldots, M_n)$:

$$\sum_{j \in N} w_j M_j = \sum_{i=1}^{n} \left(\frac{w_i}{p_i} - \frac{w_{i+1}}{p_{i+1}} \right) \sum_{j \in [i]} p_j M_j = \sum_{i=1}^{n} \left(\frac{w_i}{p_i} - \frac{w_{i+1}}{p_{i+1}} \right) \sum_{l=1}^{k(i)} \sum_{j \in S_l^i} p_j M_j. \quad (3.1)$$

3.3 A Deterministic Online Algorithm for $P|r_j|\sum w_j C_j$

Consider the following online algorithm *Non-preemptive α Scheduling* (NAS), where each job j is assigned a deterministic value of α_j:

Algorithm 7 : NAS

INPUT: *A scheduling instance $I = \{(p_i, r_i, w_i),\ i \in N\}$ which is revealed online, and a vector $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_n\}$.*

(1) Construct the preemptive LP-schedule on a single virtual machine m-times faster ($I \mapsto I_m$).

(2) At job j's α_j-point $t_j(\alpha_j)$ in the virtual machine, it enters into a FIFO queue for the m machines (job j is then scheduled the first time a machine is available after all preceding jobs in the queue have started).

From now on, whenever we refer to the LP-schedule of instance I, we mean the LP-schedule in a machine that is m times faster (or the LP-schedule of I_m). Consider job j and let $\eta_k(\alpha_j)$ denote the fraction of job k that has been completed in the LP-schedule by time $t_j(\alpha_j)$. Letting $C_j^{\boldsymbol{\alpha}}$ denote the completion time of job j in algorithm NAS when the vector $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_n\}$ is applied, we can show the following bound. Bounds of similar flavor have been frequently used in the scheduling literature (e.g., see [35, 39, 58]).

Lemma 2

$$C_j^{\alpha} \leq t_j(\alpha_j) + \sum_{k:\alpha_k \leq \eta_k(\alpha_j)} \frac{p_k}{m} + (1 - \frac{1}{m})p_j.$$

Proof The completion time of job j equals the time to enter the queue for the parallel machines plus the waiting time in queue plus the processing time of job j.

The time to enter the queue is $t_j(\alpha_j)$, which is the α_j-point of job j in the single virtual machine that is m-times faster.

The wait time in the queue can be bounded as follows: Consider all jobs that entered the queue before job j, i.e., jobs belonging to the set $\{k \neq j : \alpha_k \leq \eta_k(\alpha_j)\}$ (which are all available for processing at time $t_j(\alpha_j)$ or earlier). Then the total time that needs to be processed before job j in the m machines is at most $\sum_{k \neq j:\alpha_k \leq \eta_k(\alpha_j)} p_k$. Thus the first time that a machine will free up is at most:

$$t_j(\alpha_j) + \frac{\sum_{k \neq j:\alpha_k \leq \eta_k(\alpha_j)} p_k}{m} = t_j(\alpha_j) - \frac{p_j}{m} + \sum_{k:\alpha_k \leq \eta_k(\alpha_j)} \frac{p_k}{m},$$

which is obtained by averaging the processing times of all jobs before j. Adding up the previous term with the processing time p_j gives the result. ■

Our deterministic algorithm will perform best by taking a fixed value of α for all jobs: $\alpha_j = \alpha$, $\forall j$. The following theorem is the main result of this section; we show that our algorithm is 2.6181-competitive, which improves upon the 3.28-competitive algorithm given in [57]. Its proof is an extension of the proof of Theorem 3.3 in [35] to the parallel machine case.

Theorem 10 *Algorithm NAS is* $\max\{1 + \frac{1}{\alpha}, 2 + \alpha\}$*-competitive. In particular, for* $\alpha = \frac{\sqrt{5}-1}{2}$*, the schedule is* $\left(\frac{3+\sqrt{5}}{2}\right)$*-competitive (*$\frac{3+\sqrt{5}}{2} < 2.6181$*).*

Proof Consider a canonical set $S = \{1, \ldots, l\}$ for the fast single machine. Fix a job $j \in S$ and let $\eta_k = \eta_k(\alpha)$ represent the fraction of job k processed before $t_j(\alpha)$. By reordering the elements in S such that $t_1(\alpha) \leq \ldots \leq t_l(\alpha)$, we have that

$$t_j(\alpha) - r_{\min}(S) = \sum_{k \in S} \eta_k \frac{p_k}{m} \leq \sum_{k=j}^{l} \alpha \frac{p_k}{m} + \sum_{k=1}^{j-1} \frac{p_k}{m} = \frac{\alpha}{m} p(S) + \frac{(1-\alpha)}{m} \sum_{k=1}^{j-1} p_k. \quad (3.2)$$

Let C_j^α be the completion time of job j output by algorithm NAS. Define R to be the set of jobs such that $t_k(\alpha) < r_{\min}(S)$; note that $R \bigcap S = \emptyset$ and $R \bigcup \{1, \ldots, j\} = \{k : \alpha \leq \eta_k\}$. Thus, combining Lemma 2 with Equation (3.2) and then noting that $\alpha \frac{p(R)}{m} \leq r_{\min}(S)$, we get

$$\begin{aligned} C_j^\alpha &\leq r_{\min}(S) + \frac{\alpha}{m} p(S) + \frac{(1-\alpha)}{m} \sum_{k=1}^{j-1} p_k + \frac{1}{m} p(R) + \frac{1}{m} \sum_{k=1}^{j-1} p_k + p_j \\ &\leq (1 + \frac{1}{\alpha}) r_{\min}(S) + \frac{\alpha}{m} p(S) + \frac{(2-\alpha)}{m} \sum_{k=1}^{j-1} p_k + p_j. \end{aligned}$$

Multiplying by p_j and summing over S we get

$$\sum_{j \in S} p_j C_j^\alpha \leq (1 + \frac{1}{\alpha}) r_{\min}(S) p(S) + \frac{\alpha}{m} p(S)^2 + \frac{(2-\alpha)}{m} \sum_{j \in S} \sum_{k=1}^{j-1} p_j p_k + \sum_{j \in S} p_j^2.$$

Using the identity $\sum_{j \in S} \sum_{k=1}^{j-1} p_j p_k = \frac{1}{2} p(S)^2 - \frac{1}{2} \sum_{j \in S} p_j^2$ we obtain that for any canonical set S:

$$\begin{aligned} \sum_{j \in S} p_j C_j^\alpha &\leq (1 + \frac{1}{\alpha}) r_{\min}(S) p(S) + (2 + \alpha) \frac{p(S)^2}{2m} + \sum_{j \in S} p_j^2 \\ &\leq \max\{1 + \frac{1}{\alpha}, 2 + \alpha\} \left(p(S) \left(r_{\min}(S) + \frac{p(S)}{2m} \right) + \frac{1}{2} \sum_{j \in S} p_j^2 \right). \end{aligned}$$

Assume now that the jobs are ordered such that $w_1/p_1 \geq \cdots \geq w_n/p_n \geq w_{n+1}/p_{n+1} = 0$. Let us now bound the overall cost of the schedule using Equation (3.1) applied to instance I_m and the feasibility of $Z_R^m(I)$:

$$\begin{aligned} \sum_{j \in N} w_j C_j^\alpha &= \sum_{i=1}^{n} \left(\frac{w_i}{\frac{p_i}{m}} - \frac{w_{i+1}}{\frac{p_{i+1}}{m}} \right) \sum_{l=1}^{k(i)} \sum_{j \in S_l^i} \frac{p_j}{m} C_j^\alpha \\ &\leq \max\{1 + \frac{1}{\alpha}, 2 + \alpha\} \sum_{i=1}^{n} \left(\frac{w_i}{\frac{p_i}{m}} - \frac{w_{i+1}}{\frac{p_{i+1}}{m}} \right). \end{aligned}$$

$$\begin{aligned}
&\sum_{l=1}^{k(i)} \left(\frac{p(S_l^i)}{m} \left(r_{\min}(S_l^i) + \frac{p(S_l^i)}{2m} \right) + \frac{1}{2m} \sum_{j \in S_l^i} p_j^2 \right) \\
\leq\ & \max\{1 + \frac{1}{\alpha}, 2 + \alpha\} \sum_{i=1}^{n} \left(\frac{w_i}{\frac{p_i}{m}} - \frac{w_{i+1}}{\frac{p_{i+1}}{m}} \right) \sum_{l=1}^{k(i)} \left(\sum_{j \in S_l^i} \frac{p_j}{m} M_j + \frac{1}{2m} \sum_{j \in S_l^i} p_j^2 \right) \\
=\ & \max\{1 + \frac{1}{\alpha}, 2 + \alpha\} \sum_{i=1}^{n} \left(\frac{w_i}{\frac{p_i}{m}} - \frac{w_{i+1}}{\frac{p_{i+1}}{m}} \right) \sum_{l=1}^{k(i)} \sum_{j \in S_l^i} \frac{p_j}{m} \left(M_j + \frac{p_j}{2} \right).
\end{aligned}$$

Here, M_j denotes the mean-busy-time of job j in the LP-schedule. Applying Equation (3.1) again it follows that the previous quantity equals

$$\max\{1 + \frac{1}{\alpha}, 2 + \alpha\} \sum_{j \in N} w_j \left(M_j + \frac{p_j}{2} \right) = \max\{1 + \frac{1}{\alpha}, 2 + \alpha\} \left(Z_R^m(I) + \frac{1}{2} \sum_{j \in N} w_j p_j \right),$$

and by Lemma 1 it follows that $\sum_{j \in N} w_j C_j^\alpha \leq \max\{1 + \frac{1}{\alpha}, 2 + \alpha\} Z^m(I)$. Finally, we recall that $Z^m(I)$ is a lower bound on the optimal offline cost of $P|r_j| \sum_j w_j C_j$. ■

As in the single machine case, there are instances for which algorithm NAS gives a schedule with cost as much as twice the LP lower bound; see, for example, [35]. However, we do not know whether our analysis is tight.

3.4 A Randomized Online Algorithm for $P|r_j| \sum_j w_j C_j$

Consider the following algorithm, which we denote as *Non-preemptive α Scheduling Randomized* (NASR).

Algorithm 8 : NASR

INPUT: *A scheduling instance $I = \{(p_i, r_i, w_i),\ i \in N\}$ which is revealed online and a distribution f.*

(1) Construct the preemptive LP-schedule on a single virtual machine m-times faster ($I \mapsto I_m$).

(2) Each α_j is taken identically and independently from distribution $f(\alpha)$.

(3) At job j's α_j-point $t_j(\alpha_j)$, it enters into a FIFO queue for the m machines (job j is then scheduled the first time a machine is available after all preceding jobs in the queue have started).

We start by proving that the uniform distribution already gives 2-competitiveness; this matches the bound in [64] with a list-scheduling algorithm. The proof is very similar to Theorem 3.4 in [35].

Theorem 11 *NASR is 2-competitive when $f(\alpha)$ is the uniform distribution on $(0, 1]$.*

Proof Let $C_j^{\boldsymbol{\alpha}}$ be the completion time of job j in the schedule given by algorithm NASR. We apply Lemma 2 and first find a conditional expectation, holding α_j constant:

$$\mathrm{E}[C_j^{\boldsymbol{\alpha}} \mid \alpha_j] \leq t_j(\alpha_j) + \sum_{k \neq j} \frac{p_k}{m} \int_0^{\eta_k(\alpha_j)} d\alpha_k + p_j = t_j(\alpha_j) + \sum_{k \neq j} \frac{p_k}{m} \eta_k(\alpha_j) + p_j \leq 2(t_j(\alpha_j) + \frac{p_j}{2}).$$

This implies that

$$\mathrm{E}\left\{C_j^{\boldsymbol{\alpha}}\right\} \leq \int_0^1 2(t_j(\alpha_j) + \frac{1}{2}p_j) d\alpha_j = 2(M_j + \frac{1}{2}p_j),$$

where M_j denotes the mean-busy-time of job j in the LP-schedule. Multiplying by w_j and summing over j we get

$$\mathrm{E}\left[\sum_{j \in N} w_j C_j^{\boldsymbol{\alpha}}\right] \leq 2\left(Z_R(I_m) + \frac{1}{2}\sum_{i \in N} w_i p_i\right) \leq 2 \cdot Z^m(I),$$

which proves the result. ■

We now turn to deriving improved bounds which will depend on the number of machines. We show that by taking the α_j from an appropriate distribution we can improve on 2-competitiveness. Let us start by giving a refinement of Lemma 2.

Lemma 3

$$C_j^{\boldsymbol{\alpha}} \leq t_j(\alpha_j) + \sum_{\substack{k:\alpha_k \leq \eta_k(\alpha_j) \\ k \neq j}} \left(1 + \frac{\alpha_k}{m} - \frac{\eta_k(\alpha_j)}{m}\right) \frac{p_k}{m} + p_j.$$

Proof As in Lemma 2, the completion time of job j is equal to the time to enter the queue for the parallel machines plus the wait-time in queue plus the processing time of job j. The only difference in the bound we are attempting to prove here lies in the in-queue waiting time. This can be bounded as follows.

Consider the set K of jobs that entered the queue before job j; i.e., $K = \{k : \alpha_k \leq \eta_k(\alpha_j), k \neq j\}$. If at time $t > t_k(\alpha_k)$ the fast machine is processing job k, then at least one of the parallel machines is busy (maybe processing a different job). Thus, at time $t_j(\alpha_j)$, the parallel machines have together processed at least $\sum_{k \in K}(\eta_k(\alpha_j) - \alpha_k)\frac{p_k}{m}$. Now, the total processing requirement entered into the queue before job j is $\sum_{k \in K} p_k$. Since we have just argued that by time $t_j(\alpha_j)$, the m machines have processed $\sum_{k \in K}(\eta_k(\alpha_j) - \alpha_k)\frac{p_k}{m}$, the remaining processing requirement in the system at time $t_j(\alpha_j)$ is at most

$$\sum_{k \in K} p_k - \sum_{k \in K}(\eta_k(\alpha_j) - \alpha_k)\frac{p_k}{m} = \sum_{k \in K}\left(1 + \frac{\alpha_k}{m} - \frac{\eta_k(\alpha_j)}{m}\right) p_k.$$

Using standard averaging arguments, the first time a machine will empty up to process job j is at most $\sum_{k \in K}\left(1 + \frac{\alpha_k}{m} - \frac{\eta_k(\alpha_j)}{m}\right)\frac{p_k}{m}$. ■

For a given job j, we partition $N \setminus \{j\}$ into N_1 and N_2. N_2 is the set of all jobs that are processed between the start and completion of job j on the fast virtual machine and N_1 consists of any remaining jobs. For any $k \in N_2$, we let μ_k denote the fraction

of job j that, in the LP schedule of I_m, is processed before the start of job k. This implies $\forall k \in N_2$

$$\eta_k(\alpha_j) = \begin{cases} 0, & \alpha_j \leq \mu_k \\ 1, & \alpha_j > \mu_k. \end{cases}$$

Letting $t_j(0^+)$ denote the start time of job j, we may then write

$$t_j(\alpha_j) = t_j(0^+) + \sum_{\substack{k \in N_2 \\ \alpha_j > \mu_k}} \frac{p_k}{m} + \alpha_j \frac{p_j}{m}.$$

Recalling that in the LP-schedule $M_j = \int_0^1 t_j(\alpha) d\alpha$ we have that

$$M_j = t_j(0^+) + \sum_{k \in N_2} (1 - \mu_k) \frac{p_k}{m} + \frac{1}{2} \frac{p_j}{m}. \tag{3.3}$$

We can now rewrite Lemma 3 as

$$C_j^{\boldsymbol{\alpha}} \leq t_j(0^+) + \sum_{\substack{k \in N_1 \\ \alpha_k \leq \eta_k(\alpha_j)}} \left(1 + \frac{\alpha_k}{m} - \frac{\eta_k(\alpha_j)}{m}\right) \frac{p_k}{m} + \sum_{\substack{k \in N_2 \\ \alpha_j > \mu_k}} \left(2 - \frac{1 - \alpha_k}{m}\right) \frac{p_k}{m} + \left(1 + \frac{\alpha_j}{m}\right) p_j. \tag{3.4}$$

This bound will prove useful in the proof of the main result in this section.

For any $m \geq 1$, consider the following equation, which extends the equation in Theorem 3.9 in [35] to an arbitrary number of machines:

$$\ln\left(1 + \tfrac{1}{m} - \tfrac{\gamma}{m}\right) + \tfrac{\gamma}{m} = \frac{e^{(-\gamma/m)}\left(1 + \frac{1}{m} - \frac{\gamma}{m} - e^{(-\gamma/m)}\right)\left(me^{(\gamma/m)} - \gamma e^{(\gamma/m)} + \frac{1}{m}e^{(\gamma/m)} + 1 - m\right)}{1 + \frac{1}{m} - \frac{\gamma}{m}}. \tag{3.5}$$

For any finite value of m, it can be shown that equation (3.5) has a unique solution $\gamma \in (0, 1)$ (see Section A.1 in the Appendix). We set

$$\frac{\delta_m}{m} = \ln\left(1 + \frac{1}{m} - \frac{\gamma}{m}\right) + \frac{\gamma}{m},$$

for the unique value of γ that satisfies equation (3.5). It can also be shown that $\delta_m \in (0, 1)$ for any finite m (see Section A.1 in the Appendix). With this, we can

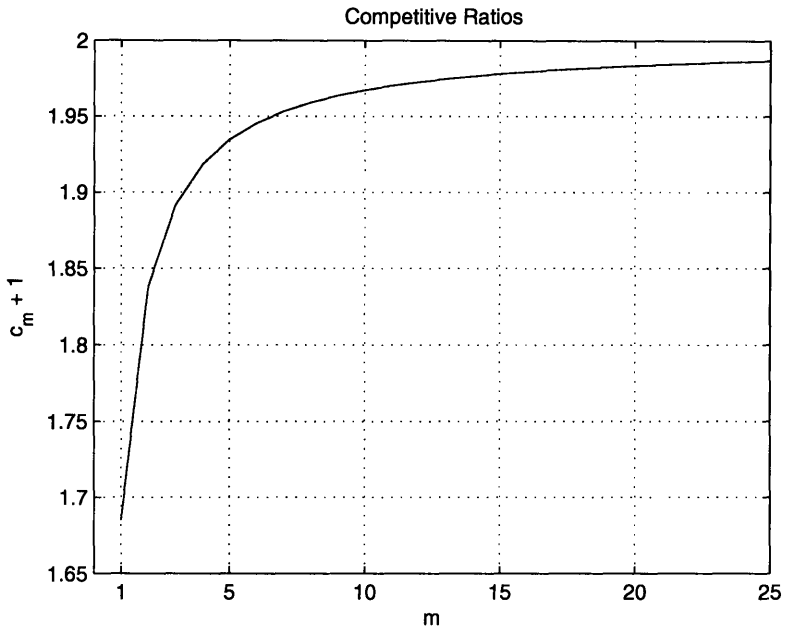

Figure 3-1: The competitive ratio bounds $1 + c_m$ as a function of m.

consider the following distribution:

$$f(\alpha) = \begin{cases} c_m e^{(\alpha/m)}, & 0 \leq \alpha \leq \delta_m \\ 0, & o.w. \end{cases}$$

where $c_m = (m\left(e^{\delta_m/m} - 1\right))^{-1}$. The main result is then the following.

Theorem 12 *With $f(\alpha)$ as above, NASR is $(1 + c_m)$-competitive.*

The competitive ratio bounds $(1 + c_m) < 2$ are plotted in Figure 3-1. Our result improves upon the 2-competitive randomized algorithms in [64]. The algorithm we present can be seen as the parallel machine extension of the algorithm in [35] for a single machine. Indeed, the competitive ratio that it achieves is 1.6853 for $m = 1$ (as in Goemans et al.); for $m = 2, 3$ and 4 it is 1.8382, 1.8915 and 1.9184, respectively. To prove this theorem, we first need a series of technical lemmas.

Lemma 4 $\int_0^\eta f(\alpha)(1 + \frac{\alpha}{m} - \frac{\eta}{m})d\alpha \leq c_m\eta, \ \forall \eta \in [0, 1].$

Proof For $\eta \in [0, \delta_m]$,

$$\begin{aligned}\int_0^\eta f(\alpha)(1+\frac{\alpha}{m}-\frac{\eta}{m})d\alpha &= c_m\left[(m-\eta)(e^{(\eta/m)}-1)\right. \\ &\quad \left.+\frac{1}{m}(m\eta e^{(\eta/m)}-m^2e^{(\eta/m)}+m^2)\right] \\ &= c_m\eta.\end{aligned}$$

For $\eta \in (\delta_m, 1]$,

$$\int_0^\eta f(\alpha)(1+\frac{\alpha}{m}-\frac{\eta}{m})d\alpha < \int_0^{\delta_m} f(\alpha)(1+\frac{\alpha}{m}-\frac{\delta_m}{m})d\alpha = c_m\delta_m < c_m\eta. \blacksquare$$

Lemma 5 $\left(2-\frac{(1-\mathrm{E}[\alpha])}{m}\right)\int_\mu^1 f(\alpha)d\alpha \le (c_m+1)(1-\mu),\ \forall \mu \in [0,1].$

Proof We first find the expected value of α; $\mathrm{E}[\alpha] = (mc_m+1)\delta_m - m$. Therefore,

$$2-\frac{(1-\mathrm{E}[\alpha])}{m} = 1-\frac{1}{m}+(mc_m+1)\frac{\delta_m}{m}.$$

We now turn to the other component. For $\mu \in [0, \delta_m]$,

$$\int_\mu^1 f(\alpha)d\alpha = c_m m\left(e^{(\delta_m/m)}-e^{(\mu/m)}\right).$$

Thus,

$$\left(2-\frac{(1-\mathrm{E}[\alpha])}{m}\right)\int_\mu^1 f(\alpha)d\alpha = \left(1-\frac{1}{m}+(mc_m+1)\frac{\delta_m}{m}\right)c_m m\left(e^{(\delta_m/m)}-e^{(\mu/m)}\right).$$

We now use Equation (3.5) and calculate the portion

$$\left(1-\frac{1}{m}+(mc_m+1)\frac{\delta_m}{m}\right)c_m. \tag{3.6}$$

Define $\Psi = 1+\frac{1}{m}-\frac{\gamma}{m}-e^{(-\gamma/m)}$ and $\Upsilon = me^{(\gamma/m)}-\gamma e^{(\gamma/m)}+\frac{1}{m}e^{(\gamma/m)}+1-m$, so that the RHS of Equation (3.5) can be written as $\frac{e^{(-\gamma/m)}\Psi\Upsilon}{(1+\frac{1}{m}-\frac{\gamma}{m})}$. Now, using the fact that $e^{(\delta_m/m)} = e^{(\gamma/m)}(1+\frac{1}{m}-\frac{\gamma}{m})$, we have that $c_m = \frac{e^{-(\gamma/m)}}{m\Psi}$.

Substituting this expression for c_m in Equation (3.6) and replacing $\frac{\delta_m}{m}$ with the

RHS of Equation (3.5), we have that

$$\begin{aligned}
\left(1-\frac{1}{m}+(mc_m+1)\frac{\delta_m}{m}\right)c_m &= \left(1-\frac{1}{m}+(\frac{e^{-(\gamma/m)}}{\Psi}+1)\frac{\delta_m}{m}\right)\frac{e^{-(\gamma/m)}}{m\Psi} \\
&= \left(1-\frac{1}{m}+(\frac{e^{-(\gamma/m)}}{\Psi}+1)\frac{e^{(-\gamma/m)}\Psi\Upsilon}{(1+\frac{1}{m}-\frac{\gamma}{m})}\right)\frac{e^{-(\gamma/m)}}{m\Psi} \\
&= \left(1-\frac{1}{m}+(e^{-(\gamma/m)}+\Psi)\frac{e^{(-\gamma/m)}\Upsilon}{(1+\frac{1}{m}-\frac{\gamma}{m})}\right)\frac{e^{-(\gamma/m)}}{m\Psi} \\
&= \left(1-\frac{1}{m}+e^{(-\gamma/m)}\Upsilon\right)\frac{e^{-(\gamma/m)}}{m\Psi} \\
&= \left(e^{-(\gamma/m)}+m\Psi\right)\frac{e^{-(\gamma/m)}}{m\Psi} \\
&= (c_m+1)e^{-(\gamma/m)}.
\end{aligned}$$

Consequently,

$$\begin{aligned}
\left(2-\frac{(1-\mathrm{E}[\alpha])}{m}\right)\int_\mu^1 f(\alpha)d\alpha &= (c_m+1)e^{-(\gamma/m)}m\left(e^{(\delta_m/m)}-e^{(\mu/m)}\right) \\
&= (c_m+1)m\left(1+\frac{1}{m}-\frac{\gamma}{m}-e^{((\mu-\gamma)/m)}\right) \\
&\leq (c_m+1)m\left(1+\frac{1}{m}-\frac{\gamma}{m}-1-\frac{(\mu-\gamma)}{m}\right) \\
&= (c_m+1)(1-\mu). \blacksquare
\end{aligned}$$

Lemma 6 $1+\dfrac{\mathrm{E}[\alpha]}{m} \leq (c_m+1)\left(\dfrac{m+1}{2m}\right)$.

Proof A simple calculation yields

$$1+\frac{\mathrm{E}[\alpha]}{m} = \frac{\delta_m e^{(\delta_m/m)}}{m(e^{(\delta_m/m)}-1)} = \frac{2m\delta_m e^{(\delta_m/m)}}{2m^2(e^{(\delta_m/m)}-1)}. \tag{3.7}$$

Turning to the RHS of the lemma, we see that

$$c_m+1 = \frac{me^{(\delta_m/m)}-m+1}{m(e^{(\delta_m/m)}-1)},$$

and consequently

$$(c_m+1)\left(\frac{m+1}{2m}\right) = \frac{m^2e^{(\delta_m/m)}-m^2+me^{(\delta_m/m)}+1}{2m^2(e^{(\delta_m/m)}-1)}. \tag{3.8}$$

Examining the numerators of Equations (3.7) and (3.8) (and noting that the denominators are positive), it is sufficient to prove $m^2e^{(\delta_m/m)}-m^2+me^{(\delta_m/m)}+1 \geq 2m\delta_m e^{(\delta_m/m)}$; we demonstrate this as follows.

$$\begin{aligned}
& m^2e^{(\delta_m/m)}-m^2+me^{(\delta_m/m)}+1-2m\delta_m e^{(\delta_m/m)} \\
= \;& (m^2+m-2m\delta_m)e^{(\delta_m/m)}-m^2+1 \\
\geq \;& (m^2+m-2m\delta_m)(1+\frac{\delta_m}{m})-m^2+1 \\
= \;& m(1-\delta_m)+(\delta_m-\delta_m^2)+(1-\delta_m^2) \\
\geq \;& 0. \blacksquare
\end{aligned}$$

Now we prove Theorem 12.

Proof of Theorem 12 First, we fix α_j. Applying Equation (3.4) and Lemma 4 we obtain:

$$\begin{aligned}
\mathrm{E}[C_j^{\boldsymbol{\alpha}} \mid \alpha_j] \;\leq\; & t_j(0^+)+\sum_{k\in N_1}\int_0^{\eta_k} f(\alpha_k)(1+\frac{\alpha_k}{m}-\frac{\eta_k}{m})\frac{p_k}{m}d\alpha_k \\
& +\sum_{\substack{k\in N_2\\ \alpha_j>\mu_k}}(2-\frac{1-\mathrm{E}[\alpha_k]}{m})\frac{p_k}{m}+(1+\frac{\alpha_j}{m})p_j \\
\leq \;& t_j(0^+)+c_m\sum_{k\in N_1}\eta_k\frac{p_k}{m}+(2-\frac{1-\mathrm{E}[\alpha]}{m})\sum_{\substack{k\in N_2\\ \alpha_j>\mu_k}}\frac{p_k}{m}+(1+\frac{\alpha_j}{m})p_j \\
\leq \;& (c_m+1)t_j(0^+)+(2-\frac{1-\mathrm{E}[\alpha]}{m})\sum_{\substack{k\in N_2\\ \alpha_j>\mu_k}}\frac{p_k}{m}+(1+\frac{\alpha_j}{m})p_j.
\end{aligned}$$

The last inequality follows from the fact that $t_j(0^+) \geq \sum_{k\in N_1}\eta_k\frac{p_k}{m}$. We now integrate

over α_j and apply Lemmas 5 and 6 to find a bound on the unconditional expectation:

$$\begin{aligned}
\mathrm{E}[C_j^{\boldsymbol{\alpha}}] &\leq (c_m+1)t_j(0^+) + (2 - \frac{1-\mathrm{E}[\alpha]}{m}) \sum_{k\in N_2} \int_{\mu_k}^{1} f(\alpha_j)\frac{p_k}{m} d\alpha_j + (1+\frac{\mathrm{E}[\alpha_j]}{m})p_j \\
&\leq (c_m+1)t_j(0^+) + (c_m+1)\sum_{k\in N_2}(1-\mu_k)\frac{p_k}{m} + (1+\frac{\mathrm{E}[\alpha_j]}{m})p_j \\
&\leq (c_m+1)t_j(0^+) + (c_m+1)\sum_{k\in N_2}(1-\mu_k)\frac{p_k}{m} + (c_m+1)\left(\frac{m+1}{2m}\right)p_j \\
&= (c_m+1)\left(t_j(0^+) + \sum_{k\in N_2}(1-\mu_k)\frac{p_k}{m} + \frac{p_j}{2m} + \frac{p_j}{2}\right) = (c_m+1)\left(M_j + \frac{p_j}{2}\right),
\end{aligned}$$

where the last equality follows from Equation (3.3). Multiplying by w_j and summing over j gives

$$\begin{aligned}
\mathrm{E}\left[\sum_{j\in N} w_j C_j^{\boldsymbol{\alpha}}\right] &\leq (c_m+1)\left(\sum_{j\in N} w_j M_j + \frac{1}{2}\sum_{j\in N} w_j p_j\right) \\
&= (c_m+1)\left(Z_R^m(I) + \frac{1}{2}\sum_{j\in N} w_j p_j\right) \\
&\leq (c_m+1)Z^m(I),
\end{aligned}$$

which proves the result. ■

The class of distributions we applied is optimal for our analysis. Essentially, equation (3.5) is a sufficient optimality condition for our distributions and analysis technique.

We can also prove that $c_m < 1$ for any finite $m \geq 1$ and $\lim_{m\to\infty} c_m = 1$ (see Section A.1 in the Appendix). Not surprisingly then, as m grows, f uniformly approaches the uniform distribution on $[0,1]$.

3.5 A Randomized Online Algorithm for $P|r_j, pmtn|\sum_j w_j C_j$

We now consider the simpler preemptive case. Consider the following algorithm *Preemptive α Scheduling Randomized* (PASR):

Algorithm 9 : PASR

INPUT: *A scheduling instance $I = \{(p_i, r_i, w_i),\ i \in N\}$ which is revealed online and a distribution f.*

(1) Construct the preemptive LP-schedule on a single virtual machine m-times faster ($I \mapsto I_m$).

(2) Draw α randomly from the distribution $f(\alpha)$.

(3) Apply preemptive list-scheduling in order of non-decreasing $t_j(\alpha)$ on the m machines.

Repeating an observation from [63], we see that at any given time, the *order* of the $t_j(\alpha)$ of already released jobs can be found, even if the actual values of $t_j(\alpha)$ are not known. As an example, consider finding the order of $t_j(\alpha)$ and $t_k(\alpha)$ at a given time t. If the values of both alpha points are known, the order is trivial. If one value is known and the other is not, clearly the known value is at most t and the unknown value is at least t. Finally, if both values are unknown, the smaller alpha point will correspond to the larger ratio of weight to processing time.

It is interesting to note that if step (2) is replaced by "Take $\alpha = 1$" the algorithm becomes a deterministic online algorithm and it coincides with the 2-competitive algorithm for $P|r_j, pmtn|\sum_j w_j C_j$ in [57]. On the other hand, if f is taken as the uniform distribution in $[0, 1]$, PASR is also 2-competitive (and this follows as a consequence of the forthcoming analysis).

Let C_j^α denote the completion time of job j in the schedule output by algorithm PASR. Consider job j. Define J as the set of jobs that start before job j in the

LP-schedule. For any $k \neq j$, let η_k denote the fraction of job k that is completed in the LP-schedule by time $t_j(0^+)$. Note that $\eta_k = 0$, $\forall k \notin J$. We also have that $t_j(0^+) \geq \sum_{k \in J} \eta_k \frac{p_k}{m}$. Now, define $K_1 = \{k \mid t_k(\alpha) < t_j(0^+)\}$ and $K_2 = \{k \mid t_j(0^+) < t_k(\alpha) < t_j(\alpha)\}$. So $K = K_1 \cup K_2$ is the set of jobs that can preempt job j. Note that jobs $k \in K_2$ preempt job j in the LP-schedule and are all processed in the interval $[t_j(0^+), t_j(\alpha)]$. Consequently, $t_j(\alpha) = t_j(0^+) + \sum_{k \in K_2} \frac{p_k}{m} + \alpha \frac{p_j}{m}$. The following bound is central to our analysis.

Lemma 7

$$C_j^\alpha \leq t_j(\alpha) + (1 - \frac{\alpha}{m})p_j + \sum_{k \in J, \eta_k \geq \alpha} (1 - \frac{\eta_k}{m})\frac{p_k}{m}.$$

Proof We first note that if the LP-schedule is busy, then at least one machine is busy in the schedule defined by PASR. Thus, by time $t_j(0^+)$, the LP-schedule will have processed a total of $\sum_{k \in K_1} \eta_k \frac{p_k}{m}$ and consequently, so will have the schedule defined by algorithm PASR.

We now make some assumptions that can only increase the completion time of job j: (1) Job j has not begun processing in the schedule defined by PASR at time $t_j(0^+)$ and (2) jobs $k \in K_2$ are released at time $t_j(0^+)$ (note that, originally, jobs in K_2 were released sometime in the interval $[t_j(0^+), t_j(\alpha)]$). While it is obvious that (1) will only increase the completion time of job j, it is not immediately clear that (2) also increases the completion time of job j. However, as we are dealing with preemptive scheduling and as job j cannot finish before $t_j(\alpha)$, making jobs in K_2 available earlier only increases the times $t_j(x)$ for any $0 \leq x \leq 1$ (since it was possible that job j found some open machine in the interval $[t_j(0^+), t_j(\alpha)]$). Thus, job j's completion time can only increase under assumption (2).

Thus, under Assumptions (1) and (2), at time $t_j(0^+)$, the amount of *available* processing that remains from $K_1 \cup K_2$ is at most $\sum_{k \in K_2} p_k + \sum_{k \in K_1} (1 - \frac{\eta_k}{m})p_k$. Since

we have m machines, by standard averaging arguments, we have that

$$\begin{aligned} C_j^{\alpha} &\leq t_j(0^+) + \sum_{k \in K_2} \frac{p_k}{m} + \sum_{k \in K_1} (1 - \frac{\eta_k}{m}) \frac{p_k}{m} + p_j \\ &= t_j(0^+) + \sum_{k \in K_2} \frac{p_k}{m} + \sum_{k \in J, \eta_k \geq \alpha} (1 - \frac{\eta_k}{m}) \frac{p_k}{m} + p_j \\ &= t_j(0^+) + \sum_{k \in K_2} \frac{p_k}{m} + \alpha \frac{p_j}{m} + (1 - \frac{\alpha}{m}) p_j + \sum_{k \in J, \eta_k \geq \alpha} (1 - \frac{\eta_k}{m}) \frac{p_k}{m} \\ &= t_j(\alpha) + (1 - \frac{\alpha}{m}) p_j + \sum_{k \in J, \eta_k \geq \alpha} (1 - \frac{\eta_k}{m}) \frac{p_k}{m}. \blacksquare \end{aligned}$$

The next lemma generalizes a result in [63].

Lemma 8 *Suppose there exists a distribution $f(\alpha)$ and a constant $\gamma_m \in (0,1)$ such that:*

- $\max_{\alpha \in [0,1]} f(\alpha) \leq 1 + \gamma_m$.
- $(1 - \frac{\eta}{m}) \int_0^{\eta} f(\alpha) d\alpha \leq \gamma_m \eta, \ \forall \eta \in [0,1]$.
- $1 - \frac{\mathrm{E}\{\alpha\}}{m} \leq \frac{1+\gamma_m}{2}$.

Then, Algorithm PASR is $(1 + \gamma_m)$-competitive.

Proof Using Lemma 7, we have that

$$\mathrm{E}[C_j^{\alpha}] \leq \mathrm{E}[t_j(\alpha)] + \mathrm{E}[(1 - \frac{\alpha}{m}) p_j] + \mathrm{E}[\sum_{k \in J, \eta_k \geq \alpha} (1 - \frac{\eta_k}{m}) \frac{p_k}{m}].$$

We bound each term individually:

$$\begin{aligned}
\mathrm{E}[t_j(\alpha)] &= \mathrm{E}[t_j(0^+) + t_j(\alpha) - t_j(0^+)] \\
&= t_j(0^+) + \int_0^1 f(\alpha)(t_j(\alpha) - t_j(0^+))d\alpha \\
&\leq t_j(0^+) + (1+\gamma_m)(M_j - t_j(0^+)) = (1+\gamma_m)M_j - \gamma_m t_j(0^+); \\
\mathrm{E}[(1-\frac{\alpha}{m})p_j] &= (1-\frac{\mathrm{E}[\alpha]}{m})p_j \leq (1+\gamma_m)\frac{p_j}{2}; \\
\mathrm{E}[\sum_{k\in J, \eta_k \geq \alpha}(1-\frac{\eta_k}{m})\frac{p_k}{m}] &= \sum_{k\in J}(1-\frac{\eta_k}{m})\frac{p_k}{m}\int_0^{\eta_k} f(\alpha)d\alpha \\
&\leq \gamma_m \sum_{k\in J}\eta_k\frac{p_k}{m} \leq \gamma_m t_j(0^+).
\end{aligned}$$

Summing the terms it follows that

$$\mathrm{E}[C_j^\alpha] \leq (1+\gamma_m)M_j - \gamma_m t_j(0^+) + (1+\gamma_m)\frac{p_j}{2} + \gamma_m t_j(0^+) = (1+\gamma_m)(M_j + \frac{p_j}{2}).$$

Multiplying by w_j and summing over j, we conclude that

$$\begin{aligned}
\mathrm{E}[\sum_{j\in N} w_j C_j^\alpha] &\leq (1+\gamma_m)\sum_{j\in N} w_j(M_j + p_j/2) \\
&= (1+\gamma_m)(Z_R^m(I) + \sum_{j\in N} w_j p_j/2) \\
&\leq (1+\gamma_m)Z^m(I). \blacksquare
\end{aligned}$$

Consider the following distribution for α for the case where we have m machines:

$$f(\alpha) = \begin{cases} \gamma_m \frac{m^2}{(m-\alpha)^2}, & \alpha \in [0, \delta_m] \\ (1+\gamma_m) & \alpha \in (\delta_m, 1], \end{cases}$$

where $\gamma_m = \frac{\delta_m(m-\delta_m)}{m-\delta_m(1-\delta_m)}$, and $\delta_m \in (0,1]$.

Note that for $m = 1$, if we let $\delta_1 = \frac{1}{2}$, then $\gamma_1 = \frac{1}{3}$ and f is exactly the distribution chosen in [63], which gives a $(1+\gamma_1) = \frac{4}{3}$-competitive algorithm. In the series of lemmas and corollaries that follow, we find an optimal value δ_m^* for all $m \geq 2$ and show that the above distribution satisfies the conditions of Lemma 8, proving the

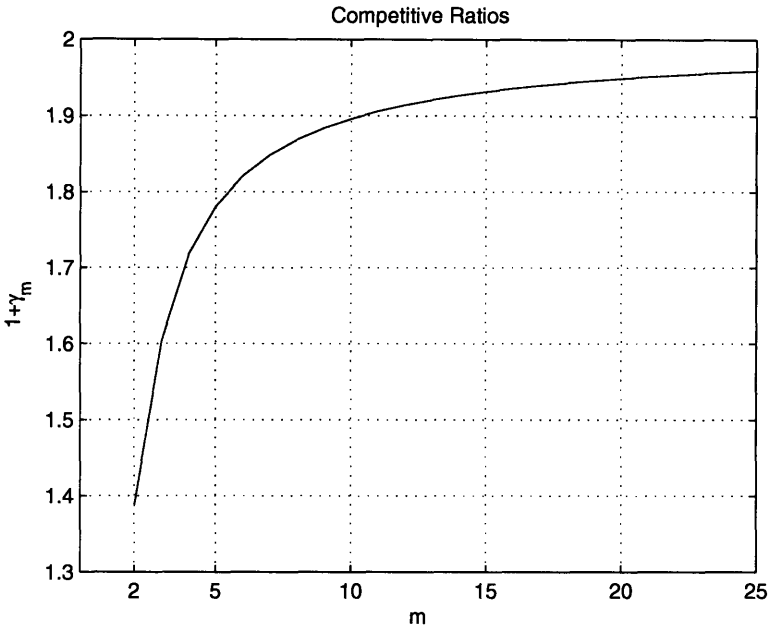

Figure 3-2: The competitive ratio bounds $1 + \gamma_m$ as a function of m.

main result of this section.

Theorem 13 *Algorithm PASR is* $(1 + \gamma_m)$*-competitive for* $m \geq 2$.

The competitive ratio bounds $(1 + \gamma_m) < 2$ are plotted in Figure 3-2. The reader may wish to compare our result with the current best algorithm to date: the deterministic algorithm in [57], which has a competitive ratio of 2 (and not better than 2) for any number of machines. Additionally, our algorithm can be simultaneously seen as an extension of the result in [57] and of the result in [63]. For two, three and four machines the competitive ratio bound is 1.3867, 1.603 and 1.7173, respectively. In general we have that $(1 + \gamma_m) < 2 - 1/m$, for $m > 1$.

Lemma 9 *For* $m \geq 2$ *and* $\forall \delta_m \in (0, \frac{m(\sqrt{m}-1)}{m-1}]$, *we have that*

$$\max_{\alpha \in [0,1]} f(\alpha) \leq 1 + \gamma_m.$$

Proof Note that on $(0, \delta_m]$, $f(\alpha)$ is increasing; consequently,

$$\max_{\alpha \in [0,\delta_m]} f(\alpha) = \gamma_m \frac{m^2}{(m-\delta_m)^2}.$$

For a contradiction, assume $\gamma_m \frac{m^2}{(m-\delta_m)^2} > 1 + \gamma_m$. This leads to $(m-1)\delta_m^2 + 2m\delta_m - m^2 > 0$, which is only true for $\delta_m > \frac{m(\sqrt{m}-1)}{m-1}$ and $\delta_m < -\frac{m(\sqrt{m}+1)}{m-1} < 0$, a contradiction. ■

Corollary 5 *For $m = 2$ and $\forall \delta_2 \in (0, 2(\sqrt{2}-1)]$, we have that*

$$\max_{\alpha \in [0,1]} f(\alpha) \leq 1 + \gamma_2.$$

Corollary 6 *For $m \geq 3$ and $\forall \delta_m \in (0, 1]$, we have that*

$$\max_{\alpha \in [0,1]} f(\alpha) \leq 1 + \gamma_m.$$

We now present a result related to the second requirement of Lemma 8.

Lemma 10 *For $m \geq 2$ and $\forall \delta_m \in (0, \frac{m(\sqrt{m}-1)}{m-1}]$, we have that*

$$(1 - \frac{\eta}{m}) \int_0^\eta f(\alpha) d\alpha \leq \gamma_m \eta, \ \forall \eta \in [0, 1].$$

Proof For $\eta \leq \delta_m$,

$$(1 - \frac{\eta}{m}) \int_0^\eta f(\alpha) d\alpha = (1 - \frac{\eta}{m}) \gamma_m m \frac{\eta}{m - \eta} = \gamma_m \eta.$$

For $\eta > \delta_m$, we first calculate

$$\int_0^\eta f(\alpha) d\alpha = \frac{m\eta - m\delta_m - \delta_m \eta + \delta_m^2 + m\delta_m \eta}{m - \delta_m(1 - \delta_m)}.$$

Multiplying the above integral by $(1 - \frac{\eta}{m})$, we have that

$$\begin{aligned} & \left(\frac{m-\eta}{m}\right) \int_0^\eta f(\alpha) d\alpha \\ = \; & \frac{(-m + \delta_m - m\delta_m)\eta^2 + (m^2 + m^2\delta_m - \delta_m^2)\eta + (-m^2\delta_m + m\delta_m^2)}{m(m - \delta_m(1 - \delta_m))}. \end{aligned}$$

Now, for a contradiction, assume $(1 - \frac{\eta}{m}) \int_0^\eta f(\alpha) d\alpha > \gamma_m \eta$; this leads to the following inequality:

$$(-m + \delta_m - m\delta_m)\eta^2 + (m^2 + m^2\delta_m - \delta_m^2)\eta + (-m^2\delta_m + m\delta_m^2) > m\delta_m(m - \delta_m)\eta,$$

or equivalently,

$$f(\eta) \triangleq (m - \delta_m + m\delta_m)\eta^2 - (m^2 + (m-1)\delta_m^2)\eta + (m^2\delta_m - m\delta_m^2) < 0.$$

Now, $f(\eta) < 0$ is possible iff its discriminant is strictly positive. The discriminant may be re-written as

$$\begin{aligned} & (m^2 + (m-1)\delta_m^2)^2 - 4(m - \delta_m + m\delta_m)(m^2\delta_m - m\delta_m^2) \\ = \; & \left((m-1)\delta_m^2 + 2m\delta_m - m^2\right)^2. \end{aligned}$$

Recall that in the proof of Lemma 9, we noted that the inequality $(m-1)\delta_m^2 + 2m\delta_m - m^2 > 0$ is valid only for $\delta_m > \frac{m(\sqrt{m}-1)}{m-1}$ and $\delta_m < -\frac{m(\sqrt{m}+1)}{m-1}$; this completes the proof. ■

Corollary 7 *For $m = 2$ and $\forall \delta_2 \in (0, 2(\sqrt{2} - 1)]$, we have that*

$$(1 - \frac{\eta}{2}) \int_0^\eta f(\alpha) d\alpha \leq \gamma_2 \eta, \; \forall \eta \in [0, 1].$$

Corollary 8 *For $m \geq 3$ and $\forall \delta_m \in (0,1]$, we have that*

$$(1 - \frac{\eta}{m}) \int_0^{\eta} f(\alpha) d\alpha \leq \gamma_m \eta, \ \forall \eta \in [0,1].$$

Finally, we discuss the third requirement of Lemma 8; here we find an optimal value of δ_m for each $m \geq 2$. Consider the following lemma.

Lemma 11 *The equation*

$$1 - \frac{\mathrm{E}[\alpha]}{m} = \frac{1+\gamma_m}{2} \tag{3.9}$$

has an unique solution $\delta_m^* \in (0,1]$ for $m \geq 2$.

Proof We first drop the subscripts m from δ_m and γ_m. We may express $\mathrm{E}[\alpha]$ as

$$\mathrm{E}[\alpha] = \gamma m^2 \left(\frac{\delta}{m-\delta} + \ln\left(\frac{m-\delta}{m}\right)\right) + (1+\gamma)\frac{1-\delta^2}{2}.$$

Noticing that $\gamma|_{\delta=0} = 0$ and $\gamma|_{\delta=1} = \frac{m-1}{m}$, we next calculate $\mathrm{E}[\alpha]$ for $\delta \in \{0,1\}$:

$$\mathrm{E}[\alpha]|_{\delta=0} = \frac{1}{2} \qquad \text{and} \qquad \mathrm{E}[\alpha]|_{\delta=1} = m + (m-1)m\ln\left(\frac{m-1}{m}\right).$$

Thus,

$$\left(1 - \frac{\mathrm{E}[\alpha]}{m}\right)\Big|_{\delta=0} = 1 - \frac{1}{2m} \qquad \text{and} \qquad \left(1 - \frac{\mathrm{E}[\alpha]}{m}\right)\Big|_{\delta=1} = (m-1)\ln\left(\frac{m}{m-1}\right).$$

We also have that

$$\left(\frac{1+\gamma}{2}\right)\Big|_{\delta=0} = \frac{1}{2} \qquad \text{and} \qquad \left(\frac{1+\gamma}{2}\right)\Big|_{\delta=1} = 1 - \frac{1}{2m}.$$

For $m \geq 2$, we have that $\left(1 - \frac{\mathrm{E}[\alpha]}{m}\right)\Big|_{\delta=0} > \left(\frac{1+\gamma}{2}\right)\Big|_{\delta=0}$. If $(m-1)\ln\left(\frac{m}{m-1}\right) \leq 1 - \frac{1}{2m}$, we can conclude that $\left(1 - \frac{\mathrm{E}[\alpha]}{m}\right)\Big|_{\delta=1} \leq \left(\frac{1+\gamma}{2}\right)\Big|_{\delta=1}$ and, consequently, there exists $\delta^* \in (0,1]$ that satisfies the lemma. We now show this latter claim. Letting $x = \frac{m}{m-1}$, (note that $m > 1 \Leftrightarrow x > 1$), we need to show that $\frac{\ln(x)}{x-1} \leq \frac{x+1}{2x}$, or equivalently,

$\ln(x) \leq \frac{x^2-1}{2x}$. Enlarging the range of x, we see that the inequality holds with equality at $x = 1$. The inequality would then be true iff the RHS increases faster than the LHS. Taking derivatives, we need to show that $\frac{1}{x} \leq \frac{x^2+1}{2x^2}$, which clearly holds for any real x.

Finally, the uniqueness of the solution δ^* can be seen by calculating a derivative. Equation (3.9) may be re-written as

$$1 = \frac{1+\gamma}{2} + \frac{\mathrm{E}[\alpha]}{m};$$

the derivative of the RHS, with respect to δ, can be calculated to be strictly positive, for any $m \geq 2$, confirming the uniqueness of δ^*. ■

The proof of Lemma 11 also gives us the following corollary.

Corollary 9 *For* $\delta_m < \delta_m^*$,

$$1 - \frac{\mathrm{E}[\alpha]}{m} > \frac{1+\gamma_m}{2}$$

and for $\delta_m > \delta_m^*$,

$$1 - \frac{\mathrm{E}[\alpha]}{m} < \frac{1+\gamma_m}{2}.$$

Note that $\delta_2^* \approx 0.4328 < 2(\sqrt{2}-1)$; thus the value of δ_2^* satisfies Corollaries 5 and 7. Consequently, Lemma 8, Corollaries 5, 6, 7, 8 and Lemma 11 give us Theorem 13.

Finally, note that for $m = 1$, $\delta_1^* = 0$, which yields a deterministic algorithm ($\alpha = 0$ with probability 1), which is not appropriate for our analysis. It has also been argued in [63] that their choice of distribution is optimal for their analysis.

We conclude this section by noting that our analysis is optimal for our approach: Since γ_m is increasing in δ_m, Corollary 9 and the third requirement of Lemma 8 show that using δ_m^* for $m \geq 2$ is the optimal choice of δ_m for the distribution $f(\alpha)$. Finally, we plot both $(1 + c_m)$ and $(1 + \gamma_m)$ for comparison in Figure 3-3.

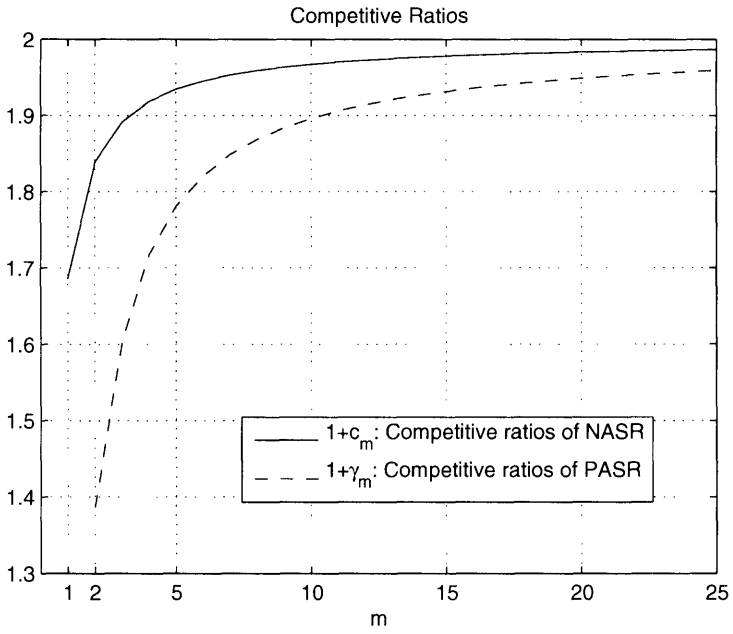

Figure 3-3: The competitive ratio bounds $1 + c_m$ and $1 + \gamma_m$ as a function of m.

Chapter 4

Resource Augmentation in Online Routing

4.1 Introduction

Resource augmentation gives the online algorithm additional power by increasing its resources. The motivation is to preclude pathological examples that give the worst-case competitive ratio; with resource augmentation, we derive improved, more realistic and meaningful competitive ratios.

We work with many types of resource augmentation: information, speed, capacity, server and constraint augmentation. Information augmentation gives the online algorithm advanced information about the problem instance. Speed augmentation gives the online servers a faster speed than the corresponding offline servers. Capacity augmentation gives the online server a larger capacity. Vehicle augmentation gives the online algorithm more servers than the offline algorithm. Constraint augmentation relaxes certain constraints for only the online algorithm.

Outline: In Section 4.2, we first consider information augmentation in the form of disclosure dates; we study the online TSP, online TSP with multiple vehicles and online TRP in a variety of metric spaces. In Section 4.3, we then study our generalized (incorporating capacity and precedence constraints) single server routing framework with speed augmentation (in Chapter 5, with additional probabilistic assumptions,

we study single server problems with both speed and capacity augmentation). In Section 4.4, we then consider multiple server routing problems with both speed and vehicle augmentation. Finally, we consider constraint augmentation for the online k-TSP in Section 4.5.

4.2 Information Resource Augmentation: Disclosure Dates

In this section, we study the online TSP, online TSP with multiple servers and the online TRP under information augmentation. We accomplish this through the introduction of city disclosure dates, times at which cities are revealed to the online algorithm, *ahead of the release dates.* Let us first state the assumptions and modified definitions about the problems we consider in the section. We detail only the single server online TSP and TRP; the multiple server online TSP is extended in a straightforward manner.

1. City locations belong to some metric space $\mathcal{M}$.

2. A city is revealed to the salesman (repairman) at its disclosure date.

3. A city is ready for service at its release date. The service requirement at a city is zero.

4. The disclosure date for a given city is less than or equal to the city's release date.

5. The salesman (repairman) travels at unit speed or is idle.

6. The problem begins at time 0, and the salesman (repairman) is initially at a designated origin of the metric space.

7. The online TSP objective is to minimize the time required to visit all cities and return to the origin.

8. The online TRP objective is to minimize the weighted sum of completion times, where each city's completion time is weighted by a given non-negative number, revealed at the city's disclosure date.

The data common to both the online TSP and online TRP is a set of points (l_i, r_i, q_i), $i = 1, \ldots, n$, where n is the number of cities. The quantity $l_i \in \mathcal{M}$ is the i^{th} city's location. The quantity $r_i \in \mathbb{R}_+$ is the i^{th} city's release date; i.e., r_i is the first time after which that city i will accept service. The quantity $q_i \in \mathbb{R}_+$ is the i^{th} city's disclosure date; i.e., at time q_i, the salesman learns about city i's request and its corresponding values l_i and r_i. We let $\mathcal{N} = \{1, \ldots, n\}$. We have that $r_i \geq q_i \geq 0$, $\forall i \in \mathcal{N}$. Finally, we let w_i, $i \in \mathcal{N}$ denote the non-negative weights on the completion times of cities for the online TRP, which become known at times q_i.

From the online perspective, the total number of requests, represented by the parameter n, is not known, and city i only becomes known at time q_i. Z_n^A will denote the cost of online algorithm A on an instance of n cities and Z_n^* is the optimal offline cost on n cities (at times, the n term will be suppressed). Finally, let $r_{max} = \max_{i \in \mathcal{N}}\{r_i\}$ and define L_{TSP} as the optimal TSP tour length through all cities in an instance.

4.2.1 The Online TSP on $\mathbb{R}^+$

In this section, we study the online TSP when the city locations are all on the non-negative real line; i.e., $\mathcal{M} = \mathbb{R}_+$. We begin with an offline analysis.

We consider the offline TSP with release dates on the non-negative real line. For this problem, [60] proposed an optimal strategy:

Algorithm 10 : Optimal Offline

(1) Go directly to city $l_{max} = \max_{i \in \mathcal{N}}\{l_i\}$.

(2) Wait at city l_{max} *for* $\max_{i \in \mathcal{N}}\{\max\{0, r_i - 2l_{max} + l_i\}\}$ *units of time.*

(3) Proceed directly back to the origin.

The waiting time is calculated to ensure the salesman's return to the origin finds each city ready for service. A closed-form expression for Z_n^* is as follows:

$$\begin{aligned} Z_n^* &= 2l_{max} + \max_{i \in \mathcal{N}} \{\max\{0, r_i - 2l_{max} + l_i\}\} \\ &= \max_{i \in \mathcal{N}} \{\max\{2l_i, r_i + l_i\}\}. \end{aligned}$$

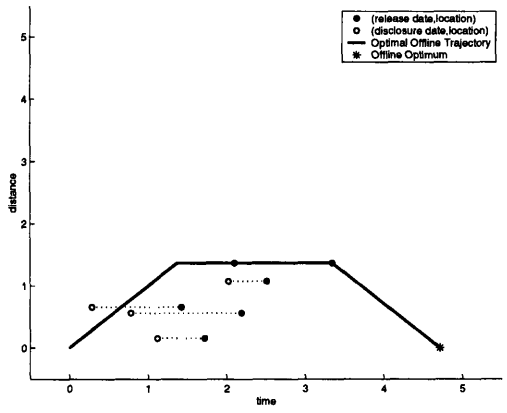

Figure 4-1: Sample trajectory of an optimal offline algorithm.

Online Algorithms

In this subsection, we consider two online algorithms. The first considers the case $q_i = r_i$, $\forall i \in \mathcal{N}$ and was first proposed and analyzed in [15], under the name of Move-Right-If-Necessary. Subsequently, we present a generalization of this algorithm for the case $q_i \leq r_i$, $\forall i \in \mathcal{N}$.

The Move-Right-If-Necessary Algorithm: We assume that $q_i = r_i$, $\forall i \in \mathcal{N}$ and we consider the following online strategy hereafter called the Move-Right-If-Necessary (MRIN) algorithm.

Algorithm 11 : MRIN

(1) If there is an unserved city to the right of the salesman, he moves towards it at unit speed.

(2) If there are no unserved cities to the right of the salesman, he moves back towards the origin at unit speed.

(3) Upon reaching the origin, the salesman becomes idle.

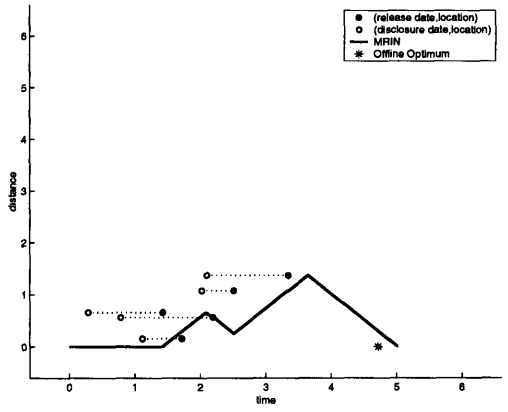

Figure 4-2: Sample trajectory of the MRIN algorithm.

The cost of the MRIN algorithm on an instance of n cities is denoted by Z_n^{MRIN}. We have the following theorem from [15].

Theorem 14 ([15]) $Z_n^{MRIN} \leq \frac{3}{2} Z_n^*, \ \forall n.$

We also have a hardness result which can be obtained from the analysis in [15].

Theorem 15 ([15]) *Let ρ be the competitive ratio for any deterministic online algorithm for the online TSP on $\mathbb{R}_+$. Then $\rho \geq \frac{3}{2}$.*

Thus, MRIN is a best-possible online algorithm (restricted to the case where $q_i = r_i, \ \forall i \in \mathcal{N}$).

The Move-Left-If-Beneficial Algorithm: We now consider the case where $q_i \leq r_i, \ \forall i \in \mathcal{N}$. Notice that by ignoring the existence of requests until their release dates, MRIN can be applied again and will yield the same competitive ratio of 3/2. However, a natural adaptation of MRIN does benefit from the disclosure dates. Thus, we define the Move-Left-If-Beneficial (MLIB) algorithm.

Algorithm 12 : MLIB

(1) If there is an unserved city to the right of the salesman, he moves towards it at unit speed.

(2) If there are no unserved cities to the right of the salesman, he moves back towards the origin if and only if the return trajectory reaches all unserved cities on or after their release date; otherwise the salesman remains idle at his current location.

(3) Upon reaching the origin, the salesman becomes idle.

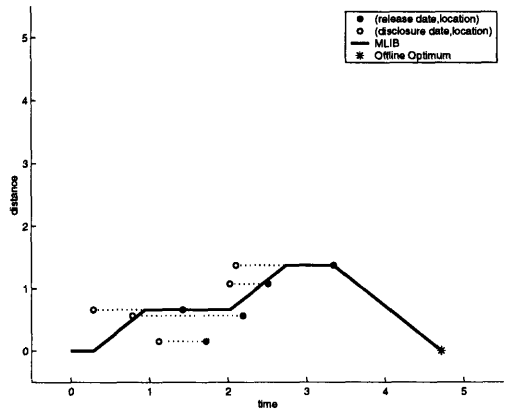

Figure 4-3: Sample trajectory of the MLIB algorithm.

The cost of the MLIB algorithm on an instance of n cities is denoted by Z_n^{MLIB}. We would like to emphasize that the MLIB algorithm applied to an instance where $q_i = r_i$, $\forall i \in \mathcal{N}$ is indistinguishable from the MRIN algorithm applied to the same instance. In addition, the MLIB algorithm applied to an instance where $q_i = 0$, $\forall i \in \mathcal{N}$ is also indistinguishable from the optimal offline algorithm. In this sense, MLIB fully incorporates the advanced information of the disclosure dates. In the next subsections, we first analyze the MLIB algorithm for a special case and then we give a general analysis.

Equal Amounts of Advanced Notice

In this subsection, we first give some technical results for the general case. Then we introduce a special structure for the disclosure dates and we show that MLIB is best-possible while MRIN is not.

Lemma 12 $Z_n^{MLIB} \leq \max_{i \in \mathcal{N}} \{\max\{q_i + 2l_i, r_i + l_i\}\}$.

Proof Suppose $Z_n^{\text{MLIB}} = z > \max_{i \in \mathcal{N}} \{\max\{q_i + 2l_i, r_i + l_i\}\}$. Consider the final segment of the MLIB salesman's trajectory; i.e., the segment of the trajectory where the salesman returns directly to the origin without changing direction or waiting. We can fully describe this segment of the trajectory as $x_t = z - t,\ t \in [t_0, z]$ for some t_0, the time the salesman begins his final return. Note that it is possible that $t_0 = z$. We have two cases to consider at time t_0:

Case (1): At t_0^-, the salesman was moving away from the origin toward a city k and reached it at t_0 such that $t_0 \geq r_k$. City k is the rightmost unserved city at time t_0 and the salesman then starts the x_t trajectory, returning to the origin, reaching each unserved city along the way on or after its release date. Since the salesman was moving away from the origin, the worst possible location for him to be when city k was disclosed was the origin. So the salesman should arrive at city k at time no later than $q_k + l_k$. Thus $x_{t_0} = l_k$, for some $t_0 \leq q_k + l_k$, implying that $z = l_k + t_0 \leq q_k + 2l_k$, which contradicts our assumption.

Case (2): The salesman has just finished waiting at some point, possibly the origin, so that the x_t trajectory reaches all cities on or after their release date. Thus, $\exists m$ such that $x_t = l_m$, for $t = r_m$, where $r_m \in [t_0, z]$. Consequently, $z = l_m + r_m$, which again contradicts our assumption. ■

We now prove a proposition that simplifies the subsequent analysis. This proposition depends on the concept of an "ignored city," which is defined as follows: An ignored city is viewed to have never existed; i.e., it will not be taken into account when calculating the online and offline costs.

Proposition 1 *For any instance of the online TSP on $\mathbb{R}_+$ that has both a request away from the origin and a request at the origin, ignoring the latter will not decrease the ratio Z_n^{MLIB}/Z_n^*.*

Proof Let $\tilde{Z}^{\text{MLIB}}$ denote the cost if the request at the origin was ignored. If the release date of the request at the origin is later than $\tilde{Z}^{\text{MLIB}}$, the proposition is trivially true. Otherwise, the behavior of MLIB is not affected by the request, but the optimal solution value may decrease by deleting it. ■

When all cities are located at the origin, we have that $Z_n^{\text{MLIB}} = Z_n^*$. The above proposition allows us to make the following assumption without a loss of generality (for our intention of proving upper bounds on competitive ratios).

Assumption 1 *$l_i > 0$ for all $i \in \mathcal{N}$.*

We now consider the situation where the online salesman receives a fixed amount of advanced notice for each city in a problem instance. In particular, there exists a constant $a \in [0, r_{max}]$ such that

$$q_i = (r_i - a)^+, \ \forall i \in \mathcal{N},$$

where $(x)^+ = \max\{x, 0\}$. Noting that $L_{TSP} = 2l_{max}$, we have the following theorem.

Theorem 16 $Z_n^{MLIB} \leq \max\left\{1, \frac{3}{2} - \alpha\right\} Z_n^*$, *where* $\alpha = \frac{a}{L_{TSP}}$.

Proof From Lemma 12, we have that

$$Z_n^{\text{MLIB}} \leq \max_{i \in \mathcal{N}} \{\max\{q_i + 2l_i, r_i + l_i\}\}. \tag{4.1}$$

Define $\mathcal{S} = \{i \in \mathcal{N} \mid q_i > 0\}$; note that for $i \in \mathcal{S}$, $q_i = r_i - a$. If $\mathcal{S} = \emptyset$, $Z_n^{\text{MLIB}} = Z_n^*$ trivially. Otherwise , we write the RHS of Equation (4.1) as

$$\max\{\max_{i \in \mathcal{S}} \{\max\{q_i + 2l_i, r_i + l_i\}\}, \max_{i \in \mathcal{N} \setminus \mathcal{S}} \{\max\{2l_i, r_i + l_i\}\}\},$$

which is less than or equal to $\max\{\max_{i\in\mathcal{S}}\{\max\{q_i+2l_i, r_i+l_i\}\}, Z_n^*\}$. Let us assume $\max_{i\in\mathcal{S}}\{\max\{q_i+2l_i, r_i+l_i\}\} > Z_n^*$; otherwise $Z_n^{\text{MLIB}} = Z_n^*$ and we are done. We can now re-write Equation (4.1) as $Z_n^{\text{MLIB}} \leq \max_{i\in\mathcal{S}}\{\max\{q_i+2l_i, r_i+l_i\}\}$. The latter term can be re-written as

$$\max_{i\in\mathcal{S}}\{r_i+l_i+\max\{(l_i-a),0\}\} \leq \max_{i\in\mathcal{S}}\{r_i+l_i+\max\{(l_{max}-a),0\}\}.$$

Now, if $a > l_{max}$, we have that $Z_n^{\text{MLIB}} \leq \max_{i\in\mathcal{S}}\{r_i+l_i\}$, which implies that $Z_n^{\text{MLIB}} = Z_n^*$, and the first part of the lemma is proved. Now, considering the case where $a \leq l_{max}$, we have that

$$\begin{aligned} Z_n^{\text{MLIB}} &\leq \max_{i\in\mathcal{S}}\{r_i+l_i+\max\{(l_{max}-a),0\}\} \\ &= \max_{i\in\mathcal{S}}\{r_i+l_i\}+(l_{max}-a) \\ &\leq Z_n^*+(l_{max}-a). \end{aligned}$$

We re-write $(l_{max}-a)$ as vl_{max}, where $v = \frac{l_{max}-a}{l_{max}} \leq 1$. Note that $vl_{max} \leq \frac{v}{2}Z_n^*$. Thus, $Z_n^{\text{MLIB}} \leq Z_n^*+(l_{max}-a) = Z_n^*+vl_{max} \leq \left(1+\frac{v}{2}\right)Z_n^* = \left(\frac{3}{2}-\frac{a}{2l_{max}}\right)Z_n^*$, and this completes the proof of the second part of the lemma. ■

Recalling that $\frac{3}{2}$ is the best-possible competitive ratio in the traditional setting, we say that the *value* of the disclosure dates is α. We now show that MLIB is in fact a best-possible algorithm in this situation.

Theorem 17 *Let A be an arbitrary deterministic online algorithm with cost Z_n^A on an instance of n cities. Then $\forall n$, there exists an instance of size n where the online cost is at least $(\frac{3}{2}-\alpha) \in [1, \frac{3}{2}]$ times the optimal offline cost, where $\alpha = a/L_{TSP}$.*

Proof We first consider the case where $n \geq 2$. Generate an instance of $(n-1)$ cities arbitrarily and let Z_{n-1}^A be the online cost of this algorithm on these $(n-1)$ cities; i.e., algorithm A serves all $(n-1)$ cities and returns to the origin at time $t = Z_{n-1}^A$. At this time, city n becomes known to algorithm A:

$$(l_n, r_n, q_n) = (a+Z_{n-1}^A, a+Z_{n-1}^A, Z_{n-1}^A).$$

Note that $l_{max} = l_n = a + Z^A_{n-1}$ since $Z^A_{n-1} \geq Z^*_{n-1} \geq 2l_i$, $\forall i < n$. Considering algorithm A, its salesman is at the origin at time q_n. Thus,

$$Z^A_n \geq q_n + 2l_n = 3Z^A_{n-1} + 2a.$$

Considering the optimal offline algorithm, we have that $Z^*_n = \max\{Z^*_{n-1}, 2(Z^A_{n-1} + a)\} = 2(Z^A_{n-1} + a)$, since $Z^A_{n-1} \geq Z^*_{n-1}$. Note that $Z^*_n > 0$ by Assumption 1. Thus,

$$\begin{aligned}
\frac{Z^A_n}{Z^*_n} &\geq 1 + \frac{Z^A_{n-1}}{2(Z^A_{n-1} + a)} \\
&= 1 + \frac{Z^A_{n-1}}{2l_{max}} \\
&= 1 + \frac{l_{max} - a}{2l_{max}} \\
&= \frac{3}{2} - \frac{a}{2l_{max}}.
\end{aligned}$$

Note that by construction, $a \leq l_{max}$ and, consequently, $\frac{3}{2} - \frac{a}{2l_{max}} \in [1, \frac{3}{2}]$.

Finally we consider the case where $n = 1$. Let $x \geq a$, $q = x - a$, $r = x$ and $l = x$. Assuming that the online algorithm does not move when no cities have been disclosed, we have that $Z^A \geq (x - a) + x + x = 3x - a$. The optimal offline cost is $Z^* = 2x$ and $Z^A/Z^* \geq 3/2 - \alpha$. ■

Notice that disclosure dates do not affect MRIN; a single city instance where $r_1 = l_1$ still induces an online cost which is $\frac{3}{2}$ times the optimal offline cost. We thus have the following corollary:

Corollary 10 *Algorithm MLIB is a best-possible online algorithm under the restriction $q_i = (r_i - a)^+$, $\forall i \in \mathcal{N}$. In addition, algorithm MRIN is not best-possible.*

In-Depth Online Analysis of MLIB Under General Disclosure Dates

In this subsection, we give a general result (of a technical nature) for the MLIB algorithm and we also present an interesting example where advanced information is actually detrimental. We first introduce some definitions:

Definition 1

1. $\delta = \min\limits_{j \in S_\delta(n)} \{\frac{q_j}{l_j}\}$, *where* $S_\delta(n) = \{j \mid q_j + 2l_j = \max_{i \in \mathcal{N}} \{\max\{q_i + 2l_i, r_i + l_i\}\}\}$.

2. $\kappa = \min\limits_{j \in S_\kappa(n)} \{\frac{l_j}{q_j}\}$, *where* $S_\kappa(n) = S_\delta(n) \bigcap \{j \mid q_j > 0\}$.

3. $\gamma = \min\limits_{j \in S_\gamma(n)} \{\frac{q_j}{r_j}\}$, *where* $S_\gamma(n) = S_\delta(n) \bigcap \{j \mid r_j > 0\}$.

Theorem 18

1. *If either or both of the sets* $S_\kappa(n)$ *and* $S_\gamma(n)$ *are empty, then* $Z_n^{MLIB} = Z_n^*$.

2. *Otherwise,* $Z_n^{MLIB} \leq (1 + \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\}) Z_n^*$.

3. *In addition, when well defined,* $(1 + \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\}) \leq \frac{3}{2}$.

Proof We first analyze the second part of the theorem, where $S_\kappa(n)$ and $S_\gamma(n)$ are both non-empty. We let m be the index that attains the minimum in the definition of δ; i.e., $q_m = \delta l_m$. By Lemma 12 and Equation (4.1), we have that

$$\begin{aligned} Z_n^{\text{MLIB}} &\leq q_m + 2l_m \\ &= (\delta + 2) l_m \\ &\leq (1 + \frac{\delta}{2}) Z_n^*. \end{aligned}$$

Let p be the index that attains the minimum in the definition of κ; i.e., $l_p = \kappa q_p$. By Lemma 12 and Equation (4.1), we have that

$$\begin{aligned} Z_n^{\text{MLIB}} &\leq q_p + 2l_p \\ &= q_p + l_p + l_p \left(\frac{1+\kappa}{1+\kappa}\right) \\ &= q_p + l_p + \frac{\kappa q_p + \kappa l_p}{1+\kappa} \\ &= (1 + \frac{\kappa}{1+\kappa})(q_p + l_p) \\ &\leq (1 + \frac{\kappa}{1+\kappa}) Z_n^*. \end{aligned}$$

Finally, we let k be the index that attains the minimum in the definition of γ; i.e., $q_k = \gamma r_k$. By Lemma 12, we have that

$$\begin{aligned} Z_n^{\text{MLIB}} &\leq q_k + 2l_k \\ &= \gamma r_k + 2l_k. \end{aligned}$$

We consider three possibilities:

(1) If $l_k > r_k$, we have that $2l_k + \gamma r_k < (2+\gamma)l_k \leq (1+\frac{\gamma}{2})Z_n^*$.

(2) If $l_k < (1-\gamma)r_k$, $2l_k + \gamma r_k < l_k + r_k \leq Z_n^*$.

(3) If $(1-\gamma)r_k \leq l_k \leq r_k$, we can let $l_k = (1-\hat{\gamma})r_k$ for some $\hat{\gamma} \in [0,\gamma]$. After some simple algebra, we see that

$$2l_k + \gamma r_k = (r_k + l_k) + \frac{(\gamma - \hat{\gamma})}{(1-\hat{\gamma})} l_k \leq Z_n^* + \gamma l_k \leq (1 + \frac{\gamma}{2}) Z_n^*,$$

where the first inequality holds because the function $f_\gamma(\hat{\gamma}) = \frac{(\gamma-\hat{\gamma})}{(1-\hat{\gamma})}$ attains a maximum of γ (when $\hat{\gamma} = 0$) on the domain $[0,\gamma]$, since $\gamma \leq 1$. Thus, $Z_n^{\text{MLIB}} \leq (1+\frac{\gamma}{2})Z_n^*$. As the previous analyses were mutually exclusive, we may conclude that, if $S_\kappa(n)$ and $S_\gamma(n)$ are both not empty, $Z_n^{\text{MLIB}} \leq (1 + \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\})Z_n^*$.

We now analyze the first part of the theorem. We have that either or both $S_\kappa(n)$ and $S_\gamma(n)$ are empty. We first consider the case where the superset $S_\delta(n) = \emptyset$. In this situation, there exists a city j s.t. $r_j + l_j = \max_{i \in \mathcal{N}} \{\max\{q_i + 2l_i, r_i + l_i\}\}$. By Lemma 12 and Equation (4.1) we have that

$$\begin{aligned} Z_n^{\text{MLIB}} &\leq r_j + l_j \\ &\leq Z_n^*. \end{aligned}$$

Recalling that $Z_n^{\text{MLIB}} \geq Z_n^*$, we conclude that $Z_n^{\text{MLIB}} = Z_n^*$. Now, assume $S_\delta(n)$ contains at least one element. If $S_\kappa(n)$ is empty, then $\delta = 0$. The analysis that results in Equation (3.9) proves that $Z_n^{\text{MLIB}} \leq Z_n^*$. Thus, $Z_n^{\text{MLIB}} = Z_n^*$. Now, if $S_\gamma(n)$ is empty, $r_j = 0$, $\forall j \in S_\delta(n)$. This again implies that $\delta = 0$ and, consequently, $Z_n^{\text{MLIB}} = Z_n^*$.

We conclude by analyzing the third part of the theorem. Since $\min\{\delta, \kappa\} \leq 1$ (also $\gamma \leq 1$), $(1 + \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\}) \leq \frac{3}{2}$. ■

Since the best-possible online algorithm, with no disclosure dates, has a competitive ratio of $\frac{3}{2}$, we say that the *value* of the disclosure dates is

$$\begin{cases} \frac{1}{2} - \min\{\frac{\gamma}{2}, \frac{\delta}{2}, \frac{\kappa}{1+\kappa}\}), & \text{if } \kappa \text{ and } \gamma \text{ are well defined} \\ \frac{1}{2}, & o.w. \end{cases}$$

To conclude our analysis of the online TSP on the non-negative real line, we provide an example where the advanced information of the disclosure dates is actually detrimental.

Example: Consider the two city instance where $q_1 = 0$, $r_1 = l_1 = 1$, $q_2 = r_2 = 2$ and $l_2 = 1$. This instance induces the following costs: $Z^{\text{MRIN}}(2) = 3$ and $Z^{\text{MLIB}}(2) = 4$.

However, we have conducted computational experiments that confirm the intuitively clear superiority of MLIB over MRIN on average; please see Chapter 6.

4.2.2 The Online TSP on $\mathcal{M}$

We now consider the general case where cities belong to a generic metric space $\mathcal{M}$. Let $d(\cdot, \cdot)$ be the metric for the space and o the origin. We consider the value of advanced information, for the structure $q_i = (r_i - a)^+$, $\forall i \in \mathcal{N}$, providing lower and upper bounds on the competitive ratio. The proof of our first result consists of simple modifications of the proof of Theorem 3.1 in Lipmann [55].

Theorem 19 *Any ρ-competitive algorithm for the online TSP on a metric space $\mathcal{M}$, with $q_i = (r_i - a)^+$, $i \in \mathcal{N}$, has $\rho \geq 2/(1 + \alpha)$, where $\alpha = a/L_{TSP}$.*

Proof Define a metric space $\mathcal{M}$ as a graph with vertex set $V = \{1, 2, \ldots, n\} \cup \{o\}$ with distance function d that satisfies the following: $d(o, i) = 1$ and $d(i, j) = 2$ for all $i \neq j \in V \setminus \{o\}$.

At time 0, there is a request at each of the n cities in $V \setminus \{o\}$. If an online server visits the request at city i at time $t \leq 2n - 1 - \epsilon$, for some small ϵ, then at time $t + \epsilon$, a new request is disclosed at city i.

In this way, at time $2n - 1$ the online server still has to serve requests at all n cities; furthermore, at time $2n - 1$, all cities have only been disclosed, not necessarily released. Therefore, the online cost is at least the corresponding value in the situation where all cities have been released by time $2n - 1$. This latter value is at least $4n - 2$. Therefore, denoting Z^A as the online cost of an arbitrary online algorithm A, we have that $Z^A \geq 4n - 2$.

The optimal offline server will also have some difficulty with the differences between the disclosure dates and release dates. We first note that, had the cities been released at the above mentioned times, rather than disclosed, the optimal offline cost would have been $2n$. We now exploit the structure of the disclosure date/release date relationship: by waiting a units of time at any disclosed city, the city's release date will arrive. Therefore, it is clear that $Z^* \leq 2n + a$. Finally, by noting that $L_{TSP} = 2n$, we have that

$$\frac{Z^A}{Z^*} \geq \frac{4n - 2}{2n + a} = \frac{2}{1 + \alpha} - \frac{2}{2n + a}.$$

Taking n arbitrarily large proves the theorem. ■

Now, we give the first of two generalizations of the 2-competitive online algorithm PAH ([10]). We call our algorithm Plan-At-Home-disclosure-dates (PAH-dd).

Algorithm 13 : PAH-dd

(1) Whenever the salesman is at the origin, it starts to follow a tour that serves all cities whose disclosure dates have passed but have not yet been served; this tour is constructed using an algorithm A that exactly solves an offline TSP with release dates.

(2) If at time q_i, for some i, a new city is presented at point l_i, the salesman takes one of two actions depending on the salesman's current position p:

(2a) If $d(l_i, o) > d(p, o)$, the salesman goes back to the origin where it appears in a Case (1) situation.

(2b) If $d(l_i, o) \leq d(p, o)$, the salesman ignores the city until it arrives at the origin, where again it re-enters Case (1).

Theorem 20 *Algorithm PAH-dd is $(2 - \frac{\alpha}{1+\alpha})$-competitive, where $\alpha = \frac{a}{L_{TSP}}$.*

Proof Let $p(t)$ be the position of the salesman at time t. Let us consider the state of the algorithm at time q_n, the final disclosure date.

Case (1): The salesman is at the origin at time q_n. Let $\mathcal{T}$ be the tour, calculated by algorithm A at time q_n, that visits all unserved cities; for simplicity, we let $\mathcal{T}$ also denote the duration of the tour. Letting $Z^{\text{PAH-dd}}$ denote the online cost of our new algorithm, we have that

$$\begin{aligned} Z^{\text{PAH-dd}} &= q_n + \mathcal{T} \\ &= r_n + \mathcal{T} - a \\ &\leq Z^* + (\mathcal{T} - a) \\ &= Z^* + (1 - \frac{a}{\mathcal{T}})\mathcal{T} \\ &\leq Z^* + (1 - \frac{a}{\mathcal{T}})Z^*, \end{aligned}$$

where the last inequality is by $\mathcal{T} \leq Z^*$. Inserting the obvious bound $\mathcal{T} \leq a + L_{TSP}$ proves the theorem for this case.

Case (2a): We have that $d(o, l_n) > d(o, p(q_n))$ and the salesman returns to the origin, arriving before time $q_n + d(o, l_n) = r_n + d(o, l_n) - a$. Once at the origin, the salesman uses algorithm A to compute a tour $\mathcal{T}'$. Clearly, $r_n + d(o, l_n) \leq Z^*$. Thus, we have that

$$Z^{\text{PAH-dd}} \leq r_n + d(o, l_n) + (\mathcal{T}' - a) \leq Z^* + (1 - \frac{\alpha}{1+\alpha})Z^* = (2 - \frac{\alpha}{1+\alpha})Z^*.$$

Case (2b): We have that $d(o, l_n) \leq d(o, p(q_n))$. Suppose that the salesman is following a route $\mathcal{R}$ that had been computed the last time he was at the origin.

Clearly, $\mathcal{R} \leq Z^*$. Let $\mathcal{Q}$ be the set of cities temporarily ignored since the last time the salesman was at the origin. Let j be the index of the first city in $\mathcal{Q}$ that is visited by the optimal offline algorithm. Let $\mathcal{P}_\mathcal{Q}$ be the shortest path starting from location l_j at time r_j, visiting all other cities in $\mathcal{Q}$, while respecting the release dates, and terminating at the origin. Clearly, $r_j + \mathcal{P}_\mathcal{Q} \leq Z^*$.

Since city j was ignored when it was disclosed, we have that $d(o, l_j) \leq d(o, p(q_j))$. Thus, at time q_j the salesman had already traveled at least a distance $d(o, l_j)$ on $\mathcal{R}$ and will complete $\mathcal{R}$ at the latest at time $t_\mathcal{R} = q_j + \mathcal{R} - d(o, l_j)$. Next, the salesman will compute $\mathcal{T}_\mathcal{Q}$, a tour covering $\mathcal{Q}$.

At time $t_\mathcal{R}$, consider an alternate strategy that first goes to city j, possibly waits for city j to be released, and then follows the shortest path through the cities in $\mathcal{Q}$; this latter path is at most $\mathcal{P}_\mathcal{Q}$. Clearly, $\mathcal{T}_\mathcal{Q}$ will finish before this alternate strategy finishes. Next, notice that the completion time of $\mathcal{T}_\mathcal{Q}$ is also the completion time of PAH-dd; therefore, we have that

$$\begin{aligned}
Z^{\text{PAH-dd}} &\leq \max\{t_\mathcal{R} + d(o, l_j), r_j\} + \mathcal{P}_\mathcal{Q} \\
&= \max\{t_\mathcal{R} + d(o, l_j) + \mathcal{P}_\mathcal{Q}, r_j + \mathcal{P}_\mathcal{Q}\} \\
&\leq \max\{t_\mathcal{R} + d(o, l_j) + \mathcal{P}_\mathcal{Q}, Z^*\} \\
&= \max\{q_j + \mathcal{R} + \mathcal{P}_\mathcal{Q}, Z^*\} \\
&= \max\{(r_j + \mathcal{P}_\mathcal{Q}) + (\mathcal{R} - a), Z^*\} \\
&\leq \max\left\{Z^* + \left(1 - \frac{\alpha}{1+\alpha}\right) Z^*, Z^*\right\} \\
&= \left(2 - \frac{\alpha}{1+\alpha}\right) Z^*. \blacksquare
\end{aligned}$$

Since the best-possible algorithm for the online metric TSP has a competitive ratio of 2, Theorems 19 and 20 indicate that the *value* of the disclosure dates is at least $\frac{\alpha}{1+\alpha}$ and no more than $\frac{2\alpha}{1+\alpha}$.

4.2.3 The Online TSP with Multiple Salesmen on $\mathcal{M}$

In this section, we investigate the value of advanced information for the multiple server case. We utilize the problem definitions from Chapter 2, Section 2.4. We add disclosure dates to the problem data in the natural way. We again consider the special case where there exists a constant $a > 0$ such that $q_i = (r_i - a)^+$ where $(x)^+ = \max\{0, x\}$. We define an appropriate algorithm to take advantage of the disclosure dates, which we denote Plan-At-Home-m-servers-disclosure-dates (PAH-m-dd).

Algorithm 14 : PAH-m-dd

(1) Whenever all servers are at the origin, they calculate and implement an exact solution to $Z^*(m)$ over all requests whose disclosure dates have passed but have not yet been served completely.

(2) If at time q_i, for some i, a new request is presented, the servers take one of two actions depending on the request's location l_i and the farthest server's current position p^* (ties broken arbitrarily):

$$p^* = \arg \max_{\{p_i \mid 1 \le i \le m\}} d(o, p_i) :$$

(2a) If $d(l_i, o) > d(p^*, o)$, all servers go back to the origin where they appear in a Case (1) situation.

(2b) If $d(l_i, o) \le d(p^*, o)$, all servers **except** *p^* return to the origin; server p^* ignores request i until it completes the route it is currently traversing, where again Case (1) is encountered.*

Theorem 21 *Algorithm PAH-m-dd is $(2 - \frac{\alpha}{1+\alpha})$-competitive, where $\alpha = a/Z^{r=0}(m)$.*

Proof Let $p^*(t)$ be the position of the farthest server at time t. Let us consider the state of the algorithm at time q_n, the final disclosure date.

Case (1): All servers are at the origin at time q_n. Letting T denote the cost of the final Case (1) re-optimization, we have that

$$\begin{aligned} Z^{\text{PAH-m-dd}} &\leq q_n + T \\ &= r_n + T - a \\ &\leq Z^*(m) + (T - a) \\ &= Z^*(m) + (1 - \frac{a}{T})T \\ &\leq Z^*(m) + (1 - \frac{a}{T})Z^*(m). \end{aligned}$$

Inserting the obvious bound $T \leq a + Z^{r=0}(m)$ proves the theorem for this case.

Case (2a): We have that $d(o, l_n) > d(o, p^*(q_n))$ and the servers return to the origin, arriving before time $q_n + d(o, l_n) = r_n + d(o, l_n) - a$. Once at the origin, the servers re-optimize; let T' denote the cost of this re-optimization. Clearly, $r_n + d(o, l_n) \leq Z^*(m)$. Thus, we have that

$$Z^{\text{PAH-dd}} \leq r_n + d(o, l_n) + (T' - a) \leq Z^*(m) + (1 - \frac{\alpha}{1+\alpha})Z^*(m) = (2 - \frac{\alpha}{1+\alpha})Z^*(m).$$

Case (2b): We have that $d(o, l_n) \leq d(o, p^*(r_n))$ and all servers, except p^*, return to the origin, if not yet already there. Suppose server p^* is following a tour $\mathcal{R}$ that had been computed the last time it was at the origin. Note that $\mathcal{R} \leq Z^*(m)$. Let $\mathcal{Q}$ be the set of requests temporarily ignored since the last time a Case (1) re-optimization was performed; since $l_n \in \mathcal{Q}$, $\mathcal{Q}$ is not empty. Let $\mathcal{S} \subseteq \{1, \ldots, m\}$ denote the set of servers that serve $\mathcal{Q}$ in the optimal offline solution. For $j \in \mathcal{S}$, let l^j be the location of the first city in $\mathcal{Q}$ served by server j in the *optimal offline solution* and let r^j be the time at which this city was released. Let $\mathcal{P}_{\mathcal{Q}}^j$, $j \in \mathcal{S}$, be the set of paths, the j-th path starting from l^j, that collectively visit all the cities in $\mathcal{Q}$ and end at the origin, such that the maximum path length is minimized. As was argued in the proof of Theorem 26, $Z^*(m) \geq \max_{j \in \mathcal{S}}\{r^j + \mathcal{P}_{\mathcal{Q}}^j\}$ and $Z^*(m) \geq \max_{j \in \mathcal{S}}\{d(o, l^j) + \mathcal{P}_{\mathcal{Q}}^j\}$.

At time q^j, the distance that salesman p^* still has to travel on the route $\mathcal{R}$ before arriving at the origin is at most $\mathcal{R} - d(o, l^j)$, since $d(o, p^*(q^j)) \geq d(o, l^j)$ implies that p^*

has traveled on the route $\mathcal{R}$ a distance not less than $d(o, l^j)$. Therefore, it will arrive at the origin before time $q^j + \mathcal{R} - d(o, l^j)$; note that since this is valid for any j, we can say that the salesman will arrive at the origin before time $\min_{j \in \mathcal{S}}\{q^j + \mathcal{R} - d(o, l^j)\}$. Note that all other salesmen have already arrived at the origin. Next, a re-optimization will be implemented on $\mathcal{Q}$; let $\mathcal{T}_{\mathcal{Q}}$ denote the maximum tour length. Hence, the completion time of PAH-m-dd will be at most $\min_{j \in \mathcal{S}}\{q^j + \mathcal{R} - d(o, l^j)\} + \mathcal{T}_{\mathcal{Q}}$. Again, $\mathcal{T}_{\mathcal{Q}} \leq \max_{j \in \mathcal{S}}\{d(0, l^j) + \mathcal{P}_{\mathcal{Q}}^j\}$ and we have that the online cost is at most

$$\min_{j \in \mathcal{S}}\{q^j + \mathcal{R} - d(o, l^j)\} + \max_{j \in \mathcal{S}}\{d(0, l^j) + \mathcal{P}_{\mathcal{Q}}^j\}.$$

Letting k be the arg max of the second term, we have that the online cost is at most

$$q^k + \mathcal{R} - d(o, l^k) + d(0, l^k) + \mathcal{P}_{\mathcal{Q}}^k = \left(r^k + \mathcal{P}_{\mathcal{Q}}^k\right) + (\mathcal{R} - a) \leq \left(2 - \frac{\alpha}{1+\alpha}\right) Z^*(m) \blacksquare$$

Theorem 22 *Any ρ-competitive algorithm with $q_i = (r_i - a)^+$, $\forall i \in \mathcal{N}$, has $\rho \geq 2/(1+\alpha)$, where $\alpha = a/Z_m^{r=0}(m)$.*

Proof Define a metric space $\mathcal{M}$ as a graph with vertex set $V = \{1, 2, \ldots, n\} \cup \{o\}$ with distance function d that satisfies the following: $d(o, i) = 1$ and $d(i, j) = 2$ for all $i \neq j \in V \setminus \{o\}$. For simplicity, assume m divides n evenly.

At time 0, there is a request at each of the n cities in $V \setminus \{o\}$. If an online server visits the request at city i at time $t \leq 2\frac{n}{m} - 1 - \epsilon$, for some small ϵ, then at time $t + \epsilon$, a new request is disclosed at city i. In this way, at time $2\frac{n}{m} - 1$ the online servers still have to serve requests at all n cities, some of which are only disclosed and not released. If all cities were released, the online servers could finish at time $(2\frac{n}{m} - 1) + 1 + 2(\frac{n}{m} - 1) = 4\frac{n}{m} - 2$; therefore this is a lower bound for the online cost when cities have only been disclosed. Denoting Z^A as the online cost of an arbitrary online algorithm A, we have that $Z^A \geq 4\frac{n}{m} - 2$. The optimal offline servers, however, will be able to visit all cities by time $2\frac{n}{m} + a$. Therefore, by letting $k = \frac{n}{m}$ and noting

that $Z_n^{r=0}(m) = 2k$, we have that

$$\frac{Z^A}{Z^*} \geq \frac{4k-2}{2k+a} = \frac{2}{1+\alpha} - \frac{2}{2k+a};$$

taking k arbitrarily large proves the theorem. ■

4.2.4 The Online TRP on $\mathcal{M}$

Thus far, we have been analyzing versions of the online TSP, where the objective is arguably in the salesman's interest. We now consider another objective, the weighted latency, which is an objective that is arguably in the cities' interest; additionally, the weights may be chosen to favor certain cities over others.

In this section, we consider the online TRP with arbitrary weights. Our objective is to minimize $\sum_{i \in \mathcal{N}} w_i C_i$, where C_i is the completion time of city i, the first time it is visited after its release date, and the w_i are arbitrary non-negative weights. Again, $l_i \in \mathcal{M}$, for any metric space $\mathcal{M}$ and we consider the situation where $q_i = (r_i - a)^+$, $\forall i \in \mathcal{N}$.

A Deterministic Online TRP Algorithm for General $\mathcal{M}$

Let $\lambda = (1 + \sqrt{2})$, $b_0 = \min\{r_j \mid r_j \geq \frac{a}{\lambda}\}$ and $b_i = \lambda^i b_0$. Also, let $\tilde{b}_i = b_i - a$. The definition of b_0 ensures that $\tilde{b}_1 \geq 0$, which is necessary for step 1 of BREAK to be feasible. The latter $\tilde{b}_i$ parameters are the breakpoints where the online algorithm BREAK (to be defined shortly) will generate some re-optimization. Our algorithm is a generalization of the $(1 + \sqrt{2})^2$-competitive INTERVAL given in [51], which re-optimizes at times b_i. Let Q_i, $i \geq 1$ denote the set of cities released up to and including time b_i; clearly $Q_i \subseteq Q_{i+1}$, $\forall i$. Note that at time $\tilde{b}_i$ the online repairman knows Q_i. Let R_i denote the set of cities served by algorithm BREAK in the interval $[\tilde{b}_i, \tilde{b}_{i+1}]$ and R_i^* the set of cities served by the optimal offline algorithm in the interval $[b_{i-1}, b_i]$. Finally, let $w(S) = \sum_{i \in S} w_i$.

Algorithm 15 : BREAK [1]

[1]Note that BREAK is not a polynomial-time algorithm since step 2 requires the exact solution

1. *Remain idle at the origin until time* $\tilde{b}_1$.

2. *At time* $\tilde{b}_1$ *calculate a path of length at most* b_1 *to serve a set of cities* $R_1 \subseteq Q_1$ *such that* $w(R_1)$ *is maximized.*

3. *At time* $\tilde{b}_i$, $i \geq 2$, *return to the origin and then calculate a path of length at most* b_i *to serve a set of cities* $R_i \subseteq Q_i \setminus \bigcup_{j<i} R_j$ *such that* $w(R_i)$ *is maximized.*

This algorithm is easily seen to be feasible – actions in iteration i are completed before actions in iteration $(i+1)$ are to begin. We begin our analysis of algorithm BREAK with the following lemma, which generalizes a result in [51]. Our proof of this lemma is quite different from that of [51] and follows the proof of a similar result in the machine scheduling literature (see [39]).

Lemma 13 *For any* $k \geq 1$, $\sum_{i=1}^{k} w(R_i) \geq \sum_{i=1}^{k} w(R_i^*)$.

Proof Consider iteration $k \geq 2$ and let $R = \bigcup_{l=1}^{k} R_l^* \setminus \bigcup_{l=1}^{k-1} R_l$. If a repairman were at the origin at time zero, he could serve all the cities in the set R by time b_k.

Now, consider an online repairman at time $\tilde{b}_k$. Suppose he knew the set R. Then by returning to the origin, taking at most b_{k-1} time units, the repairman could serve the cities in R by time $\tilde{b}_k + b_{k-1} + b_k = \tilde{b}_{k+1}$ (equality since $\alpha = (1+\sqrt{2})$). Thus, in iteration k, the repairman could serve cities of total weight $w(R)$.

Unfortunately, the repairman does not know R since the R_i^* are not known until all cities are released. However, the repairman's task is to find a subset of $S = Q_k \setminus \bigcup_{l=1}^{k-1} R_l$. Since $Q_k \supseteq \bigcup_{l=1}^{k} R_l^*$, $S \supseteq R$, and the online repairman is able to choose a subset of S to serve in iteration k of total weight at least $w(R)$, since choosing R as the subset is a feasible choice. A similar argument holds for $k = 1$.

---

of the NP-hard Orienteering Problem [16].

Now, for any k,

$$\begin{aligned} w(R_k) &\geq w(R) \\ &= \sum_{j \in \bigcup_{l=1}^{k} R_l^* \setminus \bigcup_{l=1}^{k-1} R_l} w_j \\ &= \sum_{l=1}^{k} w(R_l^*) - \sum_{j \in \left(\bigcup_{l=1}^{k} R_l^*\right) \cap \left(\bigcup_{l=1}^{k-1} R_l\right)} w_j \\ &\geq \sum_{l=1}^{k} w(R_l^*) - \sum_{j \in \bigcup_{l=1}^{k-1} R_l} w_j \\ &= \sum_{l=1}^{k} w(R_l^*) - \sum_{l=1}^{k-1} w(R_l), \end{aligned}$$

which gives the result. ■

The following corollary is evident from Lemma 13.

Corollary 11 *Suppose the optimal offline algorithm visits the last city in its tour in interval $(b_{p-1}, b_p]$ for some $p \geq 1$. Then the online algorithm BREAK will visit its last city by time $\tilde{b}_{p+1}$.*

We now give the main theorem of this section.

Theorem 23 *Algorithm BREAK is $((1+\sqrt{2})^2 - \frac{\alpha\beta}{\alpha+\beta})$-competitive, where $\alpha = \frac{a}{L_{TSP}}$ and $\beta = \frac{a}{r_{max}}$.*

Proof of Theorem 23 We begin by stating Lemma 6 from [51]: Let $a_i, b_i \in \mathbb{R}_+$, for $i = 1, \ldots, p$. If $\sum_{i=1}^{p} a_i = \sum_{i=1}^{p} b_i$ and $\sum_{i=1}^{p'} a_i \geq \sum_{i=1}^{p'} b_i$ for all $1 \leq p' \leq p$, then $\sum_{i=1}^{p} \tau_i a_i \leq \sum_{i=1}^{p} \tau_i b_i$ for any non-decreasing sequence $0 \leq \tau_1 \leq \ldots \leq \tau_p$. Applying

this lemma, we have that

$$\begin{aligned} Z^{\text{BREAK}} &\leq \sum_{k=1}^{p} \tilde{b}_{k+1} w(R_k) \\ &\leq \sum_{k=1}^{p} \tilde{b}_{k+1} w(R_k^*) \\ &= \sum_{k=1}^{p} (b_{k+1} - a) w(R_k^*) \\ &= \sum_{k=1}^{p} (\lambda^2 b_{k-1} - a) w(R_k^*) \\ &= \sum_{k=1}^{p} \sum_{l \in R_k^*} (\lambda^2 b_{k-1} - a) w_l \\ &\leq \sum_{k=1}^{p} \sum_{l \in R_k^*} (\lambda^2 C_l^* - a) w_l, \end{aligned}$$

where C_l^* the the completion time of city l by the optimal offline algorithm. Now, suppose there exists γ such that $(\lambda^2 C_l^* - a) \leq \gamma C_l^*$, $\forall l$. Then, algorithm BREAK would be γ-competitive. It is clear to see that $\gamma = \lambda^2 - \frac{a}{C_{max}^*}$ is the smallest such value to satisfy the requirements, where $C_{max}^* = \max_{i \in \mathcal{N}} \{C_i^*\}$. Thus, algorithm BREAK is $(\lambda^2 - \frac{a}{C_{max}^*})$-competitive. Finally, using the fact that $C_{max}^* \leq r_{max} + L_{TSP}$, we achieve the result. ■

Since the best deterministic algorithm to date (INTERVAL) for the online metric TRP is $(1 + \sqrt{2})^2$-competitive, we say that the value of the disclosure dates is $\frac{\alpha\beta}{\alpha+\beta}$.

A Randomized Online TRP Algorithm for General $\mathcal{M}$

We may also define a randomized algorithm BREAK-R as algorithm BREAK with the following substitution: $b_0 \mapsto \lambda^U b_0$, where U is a uniform random variable on $[0, 1]$. We have the following theorem for this randomized algorithm; its proof is quite similar to that of Theorem 23 and is omitted.

Theorem 24 *Algorithm BREAK-R is* $(\Theta - \frac{\alpha\beta}{\alpha+\beta})$*-competitive, where* $\alpha = \frac{a}{L_{TSP}}$, $\beta = \frac{a}{r_{max}}$ *and* $\Theta \approx 3.86$.

Remark 1 *To the best of our knowledge, algorithms INTERVAL and RANDINTERVAL ([51]) are the best online algorithms to-date for the online TRP, regardless of the metric space; i.e., we are not aware of any algorithms that improve these results for any simpler metric spaces, such as $\mathbb{R}_+$ or $\mathbb{R}$. Therefore, we do not have any new results specific to these particular metric spaces.*

A Correction to a Previously Published Result

When $a = 0$, algorithm BREAK-R corresponds to a realization of the λ-parameterized online algorithm RANDINTERVAL, given in [51] and summarized in Chapter 2. The values of λ for which RANDINTERVAL is a feasible algorithm were given incorrectly in [51]: The given range $\lambda \in [1 + \sqrt{2}, 3]$ should have read $\lambda \in (1, 1 + \sqrt{2}]$ since the algorithm requires $\frac{(\lambda+1)}{\lambda(\lambda-1)} \geq 1$. This led to an erroneous result that stated that RANDINTERVAL was $\tilde{\Theta}$-competitive, $\tilde{\Theta} \approx 3.64$. Using the correct range for λ, it is straightforward to see that RANDINTERVAL is Θ-competitive, where $\Theta \approx 3.86$. These corrections were originally reported in [43] and are remedied in [52].

A Final Note on the Online Dial-A-Ride Problem

Finally, note that Theorems 23 and 24 also hold for the online Dial-A-Ride problem, which is a generalization of the TRP. Instead of a customer (city) requesting a visit, a customer requests a ride from a source location to a destination location. The completion time of a customer is the time that the customer reaches the destination. The subroutine in the BREAK algorithm that calculates paths maximizing the weight of served customers must simply be modified to incorporate the new requirements of a customer.

4.3 Single Server Resource Augmentation

We now return to the setup of Chapter 2, Section 2.3, where we consider the online TSP with precedence and capacity constraints. In this section, we present a result for algorithm PAH-G, which was defined in Chapter 2, under speed augmentation.

The online algorithm has a single server with a speed $\gamma \geq 1$. The offline algorithm has a single server of unit speed. Online and offline servers have identical capacities Q.

Theorem 25 *Algorithm PAH-G is* $(1 + (2\rho - 1)/\gamma)$*-competitive.*

Proof Recall that r_n is the time of the last request and $l_n^* = \arg\max_{l_n^j \mid 1 \leq j \leq k(n)} d(o, l_n^j)$. We show that in each of the Cases (1), (2a) and (2b), PAH-G is $(1 + (2\rho - 1)/\gamma)$-competitive.

In Case (1) PAH-G is at the origin at time r_n. It starts traversing a ρ-approximate set of tours that serve all the unserved requests. Since the online server has speed γ, the time needed by PAH-G is at most $r_n + \rho Z_n^{r=0}(Q)/\gamma \leq (1 + \rho/\gamma) Z_n^*(Q)$.

Considering Case (2a), we have that $d(o, l_n^*) > d(o, p)$. Then PAH-G goes back to the origin, where it will arrive before time $r_n + \frac{d(o,l_n^*)}{\gamma}$. After this, PAH-G computes and follows a ρ-approximate set of tours through all the unserved requests. Therefore, the online cost is at most $r_n + \frac{d(o,l_n^*)}{\gamma} + \rho Z_n^{r=0}(Q)/\gamma$. Noticing that $r_n + d(o, l_n^*) \leq Z_n^*(Q)$ and $2d(o, l_n^*) \leq Z_n^{r=0}(Q)$, we have that the online cost is at most

$$\begin{aligned} r_n + \frac{d(o, l_n^*)}{\gamma} + \rho \frac{Z_n^{r=0}(Q)}{\gamma} &\leq Z_n^*(Q) + (\frac{1}{\gamma} - 1) d(o, l_n^*) + \frac{\rho}{\gamma} Z_n^*(Q) \\ &\leq \left(1 + \left(\frac{2\rho - \gamma + 1}{2\gamma}\right)\right) Z_n^*(Q) \end{aligned}$$

Finally, we consider Case (2b), where $d(o, l_n^*) \leq d(o, p)$. Suppose PAH-G is following a route $\mathcal{R}$ that had been computed the last time step (1) of PAH-G had been invoked. $\mathcal{R}$ will also denote the actual distance of the route; we have that $\mathcal{R} \leq \rho Z^{r=0}(Q) \leq \rho Z^*(Q)$. Let $\mathcal{S}$ be the set of requests that have been temporarily ignored (from step (2b) of algorithm PAH-G) since the last time PAH-G invoked step (1). Let l_f be the first location of the first request in $\mathcal{S}$ visited by the offline algorithm, and let r_f be the time at which request f was released. Let $\mathcal{P}_{\mathcal{S}}^*$ be the fastest route that starts at l_f, visits all cities in $\mathcal{S}$ and ends at the origin, respecting precedence and capacity constraints. Clearly, $Z^*(Q) \geq r_f + \mathcal{P}_{\mathcal{S}}^*$ and $Z^*(Q) \geq d(o, l_f) + \mathcal{P}_{\mathcal{S}}^*$.

At time r_f, the time that PAH-G still has left to complete route $\mathcal{R}$ is at most

$(\mathcal{R}-d(o,l_f))/\gamma$, since $d(o,p(r_f)) \geq d(o,l_f^*) \geq d(o,l_f)$ implies that PAH-G has traveled on the route $\mathcal{R}$ a distance not less than $d(o,l_f)$. Therefore, the server will complete the route $\mathcal{R}$ before time $r_f + (\mathcal{R} - d(o,l_f))/\gamma$. After that it will follow a ρ-approximate set of tours that covers the set $\mathcal{S}$ of yet unserved requests; let $\mathcal{T}_{\mathcal{S}}$ denote the cost of the *optimal* set of tours. Hence, the total time to completion will be at most $r_f + (\mathcal{R} - d(o,l_f))/\gamma + \rho\mathcal{T}_{\mathcal{S}}/\gamma$. Since $\mathcal{T}_{\mathcal{S}} \leq d(0,l_f) + \mathcal{P}_{\mathcal{S}}^*$, we have that the online cost is at most

$$\begin{aligned}
& r_f + \frac{\mathcal{R} - d(o,l_f)}{\gamma} + \frac{\rho}{\gamma} d(0,l_f) + \frac{\rho}{\gamma}\mathcal{P}_{\mathcal{S}}^* \\
= \; & (r_f + \mathcal{P}_{\mathcal{S}}^*) + \frac{1}{\gamma}\mathcal{R} + \left(\frac{\rho-1}{\gamma}\right) d(0,l_f) + \left(\frac{\rho}{\gamma} - 1\right)\mathcal{P}_{\mathcal{S}}^* \\
\leq \; & Z^*(Q) + \frac{\rho}{\gamma} Z^{r=0}(Q) + \left(\frac{\rho-1}{\gamma}\right) Z^*(Q) \\
\leq \; & \left(1 + \frac{2\rho-1}{\gamma}\right) Z^*(Q).
\end{aligned}$$

Since $\rho, \gamma \geq 1$, $\max\left\{1 + \frac{\rho}{\gamma}, 1 + \frac{2\rho-1}{\gamma}, 1 + \left(\frac{2\rho-\gamma+1}{2\gamma}\right)\right\} = 1 + \frac{2\rho-1}{\gamma}$ and the theorem is proved. ■

Corollary 12 *If we use an exact algorithm in step(1) for calculating an optimal offline $Z^{r=0}(Q)$, Algorithm PAH-G is $\left(1 + \frac{1}{\gamma}\right)$-competitive.*

4.4 Multiple Server Resource Augmentation

We continue our follow up from Chapter 2, but now consider Section 2.4, where we consider the online TSP with multiple salesmen (without precedence and capacity constraints). In this section, we present a result for algorithm PAH-m, which was defined in Chapter 2, under speed and vehicle augmentation.

We give the online algorithm m identical servers, each with a speed $\gamma \geq 1$. The offline algorithm has a single server of unit speed.

Next, we make an observation: If all the city locations are closely clustered, there is little benefit to using multiple vehicles. Therefore, we make an assumption that allows

us to circumvent this fact; we then prove a useful lemma that uses this assumption.

Assumption 2 *There exists $\beta > 0$ such that for all $i, j \in \{0, \ldots, n\}$, $i \neq j$,*

$$d(l_i, l_j) \geq \beta,$$

where l_0 denotes the origin.

Lemma 14 *Under Assumption 2, $Z^{r=0}(m) \leq Z^{r=0}(1) - (m-1)\beta$.*

Proof We consider a feasible solution to $Z^{r=0}(m)$: For servers $1, \ldots, m-1$, we assign them each the $(m-1)$ locations closest to the origin. For vehicle m, we assign it the remaining $n - m + 1$ locations; clearly $Z^{r=0}(m)$ equals the distance traveled by server m. Since distances between locations are at least β, server m will require at most a distance of $L_{TSP} - (m-1)\beta$ to serve the remaining locations. ■

Next, we define a measure of the value of the lower bound β. Let

$$\phi = \frac{\beta}{L_{TSP}};$$

note that, since $L_{TSP} \geq (n+1)\beta$, we have $\phi \leq \frac{1}{n+1}$. There exist instances such that this last inequality is tight, the most obvious being the metric space where $d(l_i, l_j) = \beta$ for all $i \neq j$. We now present and prove the main results of this section.

Theorem 26 *Under Assumption 2, the competitive ratio of Algorithm PAH-m is at most*

$$1 + \frac{\rho}{\gamma}(1 - (m-1)\phi) + \frac{\rho - 1}{\gamma}.$$

Proof Let r_n be the time of the last request, l_n the position of this request and $p^*(t)$ the location of the farthest salesman at time t.

Case (1): All salesmen are at the origin at time r_n. Then they start implementing a ρ-approximate solution to $Z^{r=0}(m)$ that serves all the unserved requests. Applying

Lemma 14, the time needed by PAH-m is at most

$$r_n+\frac{\rho}{\gamma}Z^{r=0}(m) \leq Z^*(1)+\frac{\rho}{\gamma}\left(Z^{r=0}(1)-(m-1)\beta)\right) \leq \left(1+\frac{\rho}{\gamma}\left(1-(m-1)\phi\right)\right) Z^*(1).$$

Case (2a): We have that $d(o,l_n) > d(o,p^*(r_n))$. All salesmen return to the origin, where they will all arrive before time $r_n + d(o,l_n)/\gamma \leq r_n + d(o,l_n)$. After this, PAH-m computes and follows a ρ-approximate solution to $Z^{r=0}(m)$ through all unserved requests. Therefore, the online cost is at most $r_n + d(o,l_n) + \frac{\rho}{\gamma}Z^{r=0}(m)$. Noticing that $r_n + d(o,l_n) \leq Z^*(1)$ and applying Lemma 14, we have that the online cost is at most $\left(1+\frac{\rho}{\gamma}\left(1-(m-1)\phi\right)\right) Z^*(1)$.

Case (2b): We have that $d(o,l_n) \leq d(o,p^*(r_n))$ and all salesmen, except p^*, return to the origin, if not yet already there. Suppose salesman p^* is following a tour $\mathcal{R}$ that had been computed the last time it was at the origin. Note that $\mathcal{R} \leq \rho Z^{r=0}(m)$ and $Z^{r=0}(m) \leq Z^*(m) \leq Z^*(1)$. Let $\mathcal{Q}$ be the set of requests temporarily ignored since the last time a Case (1) re-optimization was performed; since $l_n \in \mathcal{Q}$, $\mathcal{Q}$ is not empty. Let $\mathcal{S} \subseteq \{1,\ldots,m\}$ denote the set of salesmen that serve $\mathcal{Q}$ in the optimal offline solution. For $j \in \mathcal{S}$, let l^j be the location of the first city in $\mathcal{Q}$ served by server j in the *optimal offline solution* and let r^j be the time at which this city was released. Let $\mathcal{P}^j_{\mathcal{Q}}$, $j \in \mathcal{S}$, be the set of paths, the j-th path starting from l^j, that collectively visit all the cities in $\mathcal{Q}$ and end at the origin, such that the maximum path length is minimized (ties broken arbitrarily). It is easy to see that $Z^*(m) \geq \max_{j\in\mathcal{S}}\{\mathcal{P}^j_{\mathcal{Q}}\}$ since the min-max-path optimization has distinct advantages over the offline solution: (1) having the servers start at cities l^j, (2) needing to only serve the cities in $\mathcal{Q}$ and (3) ignoring release dates. If the servers start from the origin, the earliest time that server j can visit city l^j is $\max\{r^j, d(0,l^j)\}$; by extension we have that $Z^*(m) \geq \max_{j\in\mathcal{S}}\{r^j + \mathcal{P}^j_{\mathcal{Q}}\}$ and $Z^*(m) \geq \max_{j\in\mathcal{S}}\{d(o,l^j) + \mathcal{P}^j_{\mathcal{Q}}\}$.

At time r^j, the distance that salesman p^* still has to travel on the route $\mathcal{R}$ before arriving at the origin is at most $\mathcal{R} - d(o,l^j)$, since $d(o,p^*(r^j)) \geq d(o,l^j)$ implies that p^* has traveled on the route $\mathcal{R}$ a distance not less than $d(o,l^j)$. Therefore, it will arrive at the origin before time $r^j + \frac{\mathcal{R}-d(o,l^j)}{\gamma}$; note that since this is valid for any j, we

can say that the salesman will arrive at the origin before time $\min_{j\in\mathcal{S}}\{r^j + \frac{\mathcal{R}-d(o,l^j)}{\gamma}\}$. Note that all other salesmen have already arrived at the origin. Next, a ρ-approximate $Z^{r=0}(m)$ will be implemented on $\mathcal{Q}$; let $\mathcal{T}_{\mathcal{Q}}$ denote the *optimal* maximum tour length. Hence, the completion time of PAH-m will be at most $\min_{j\in\mathcal{S}}\{r^j + \frac{\mathcal{R}-d(o,l^j)}{\gamma}\} + \frac{\rho}{\gamma}\mathcal{T}_{\mathcal{Q}}$. Now, note the following feasible solution for the final case (1) re-optimization: Use only the set of salesmen $\mathcal{S}$, force salesman j to first go to city l^j and then traverse path $\mathcal{P}^j_{\mathcal{Q}}$. Therefore, $\mathcal{T}_{\mathcal{Q}} \leq \max_{j\in\mathcal{S}}\{d(0,l^j) + \mathcal{P}^j_{\mathcal{Q}}\}$ and we have that the online cost is at most

$$\min_{j\in\mathcal{S}}\left\{r^j + \frac{\mathcal{R} - d(o,l^j)}{\gamma}\right\} + \rho \max_{j\in\mathcal{S}}\left\{\frac{d(0,l^j) + \mathcal{P}^j_{\mathcal{Q}}}{\gamma}\right\}.$$

Letting k be the arg max of the second term, we have that the online cost is at most

$$\begin{aligned}
& r^k + \frac{\mathcal{R} - d(o,l^k) + \rho\left(d(0,l^k) + \mathcal{P}^k_{\mathcal{Q}}\right)}{\gamma} \\
= \;& \left(r^k + \mathcal{P}^k_{\mathcal{Q}}\right) + \frac{1}{\gamma}\mathcal{R} + \left(\frac{\rho-1}{\gamma}\right) d(o,k^k) + \left(\frac{\rho}{\gamma} - 1\right)\mathcal{P}^k_{\mathcal{Q}} \\
\leq \;& Z^*(m) + \frac{\rho}{\gamma}Z^{r=0}(m) + \left(\frac{\rho-1}{\gamma}\right)\left(d(o,l^k) + \mathcal{P}^k_{\mathcal{Q}}\right) \\
\leq \;& \frac{\rho}{\gamma}\left(Z^{r=0}(1) - (m-1)\beta\right) + \left(\frac{\rho-1+\gamma}{\gamma}\right) Z^*(1) \\
\leq \;& \left(1 + \frac{\rho}{\gamma}(1 - (m-1)\phi) + \frac{\rho-1}{\gamma}\right) Z^*(1).\blacksquare
\end{aligned}$$

Corollary 13 *Under Assumption 2, if we use an exact algorithm in step(1) for calculating an optimal offline $Z^{r=0}(m)$, Algorithm PAH-m is $\left(1 + \frac{1}{\gamma}(1 - (m-1)\phi)\right)$-competitive.*

A stronger (though dependent on n) parallel development is also possible. We first give an improvement to Lemma 14.

Lemma 15 *Under Assumption 2, $Z^{r=0}(m) \leq Z^{r=0}(1) - \frac{n(m-1)}{m}\beta$*

Proof For simplicity, assume $n/m \in \mathbb{N}$. Let $l = n(m-1)/m$. Note that l satisfies the following equation: $\frac{l}{m-1} \leq n - l$. Next, we assign $l/(m-1)$ locations to each of the first $(m-1)$ servers and then assign $(n-l)$ to server m. We do this in

the following way: Pick the $l/(m-1)$ locations that form the shortest tour and assign this tour to server 1. Out of the remaining locations, find the next $l/(m-1)$ locations that form the shortest tour and assign to server 2. Repeat until vehicle $(m-1)$. Therefore, vehicle m will have the longest tour of all vehicles. Consequently, $Z^{r=0}(m) \leq Z^{r=0}(1) - l\beta = Z^{r=0}(1) - \frac{n(m-1)}{m}\beta$. ■

Using this refinement, we have the following result.

Theorem 27 *Under Assumption 2, the competitive ratio of Algorithm PAH-m is at most*

$$1 + \frac{\rho}{\gamma}\left(1 - \frac{n(m-1)}{m}\phi\right) + \frac{\rho - 1}{\gamma}.$$

Corollary 14 *Under Assumption 2, if we use an exact algorithm in step(1) for calculating an optimal offline $Z^{r=0}(m)$, Algorithm PAH-m is $\left(1 + \frac{1}{\gamma}\left(1 - \frac{n(m-1)}{m}\phi\right)\right)$-competitive.*

4.4.1 Pure Speed Augmentation

Theorem 28 *If both online and offline algorithms have m servers, the competitive ratio of Algorithm PAH-m is at most*

$$1 + \frac{2\rho - 1}{\gamma}.$$

Proof Repeat the proof of Theorem 26 but replace all instances of $Z^*(1)$ with $Z^*(m)$ (all bounds remain valid) and do not apply Lemma 14. ■

Corollary 15 *If we use an exact algorithm in step(1) for calculating an optimal offline $Z^{r=0}(m)$, Algorithm PAH-m is $\left(1 + \frac{1}{\gamma}\right)$-competitive.*

4.5 Resource Augmentation for the Online k-TSP

We now return to Chapter 2, Section 2.6, where we presented an online version of the k-TSP, which deals with finding a subset of size k of the n cities such that the makespan is minimized.

4.5.1 Constraint Augmentation

We consider constraint resource augmentation. In this case, the online server is required to visit k' cities while the offline server must visit k cities, where $k' < k$. We prove a result of the form $Z^A \leq \rho Z^* - \kappa$, where the improvement of resource augmentation is in the κ term.

Recall that Algorithm WaG, defined in Chapter 2, has a competitive ratio of 2. We have the following result, where $Z_{k'}^{WaG}$ is the online cost for k' cities and Z_k^* is the optimal offline cost for k cities:

Theorem 29 *Under Assumption 2,*

$$Z_{k'}^{WaG} \leq 2Z_k^* - 2(k-k')\beta.$$

Proof Sketch Using algorithm WaG, we clearly have $Z_{k'}^{WaG} \leq 2Z_{k'}^*$. Considering the offline problems with k' and k cities, we note that the second server must serve $(k-k')$ additional cities; each additional city adds at least β travel time. Therefore,

$$Z_k^* \geq Z_{k'}^* + (k-k')\,\beta,$$

and we attain our result. ■

4.5.2 Speed Augmentation

We study the case where the online server has speed $\gamma \geq 1$, the offline server has unit speed and both the online and offline servers must visit k cities.

Theorem 30 *Under Assumption 2,*

$$Z_k^A \leq 2Z_k^* - (k+1)\beta\left(1 - \frac{1}{\gamma}\right).$$

Proof Sketch At time $t = Z^*$, the online server begins implementing the offline solution. Note that all release dates of cities to be visited have passed. The slower

offline server must travel at least a distance of $(k+1)\beta$ and takes at least $(k+1)\beta$ units of time to do this. Considering the fast online server, it must travel at least the same distance, but since it travels at speed γ, it will take at least $(k+1)\beta/\gamma$ units of time. Therefore, from time t, the online server will have an advantage of at least $(k+1)\beta\left(1-\frac{1}{\gamma}\right)$. We let L_γ and L denote the time required for the online and offline servers, respectively, to visit the chosen cities. Consequently, we have that

$$Z_k^{WaG} = t + L_\gamma \leq t + L - (k+1)\beta\left(1-\frac{1}{\gamma}\right) \leq 2Z_k^* - (k+1)\beta\left(1-\frac{1}{\gamma}\right). \blacksquare$$

4.5.3 Combining Speed and Constraint Augmentation

The nature of the two above results allows us to simply combine them.

Corollary 16 *Under Assumption 2,*

$$Z_{k'}^{WaG} \leq 2Z_k^* - \left((k'+1)\left(1-\frac{1}{\gamma}\right) + 2(k-k')\right)\beta.$$

Chapter 5

Stochastic Asymptotic Analysis

5.1 Introduction

In this chapter, we study online optimization problems from a new perspective: We introduce stochastic structure to the problem data. However, the online algorithms are unaware of this structure and do not use this information in any way; stated differently, we use these stochastic assumptions as a tool to study the behavior of online algorithms in a novel manner. We exclusively utilize the asymptotic competitive ratio and the corresponding notion of an algorithm being asymptotically competitive. If random variables Z_n^A and Z_n^* are the costs of an online algorithm A and an optimal offline algorithm, respectively, on an instance of size n (e.g., the number of cities in the online TSP), we say that A is almost surely asymptotically c-competitive if there exists n_0 such that for all $n \geq n_0$, $Z_n^A/Z_n^* \leq c$, almost surely. The majority of the results in this chapter show that, under certain stochastic assumptions, there exist algorithms A where $c = 1$, almost surely; i.e., we can show that there exist algorithms that are almost surely asymptotically optimal.

We also characterize the rate of convergence to optimality. Knowing that the ratio ρ_n of the online algorithm's cost to that of the optimal offline algorithm satisfies $(\rho_n - 1) \to 0$, almost surely, we then find the smallest value α, dependent on the stochastic model, distribution and algorithm utilized, such that $n^\alpha(\rho_n - 1) \to Z$,

almost surely[1], where Z is either a non-zero constant or non-degenerate random variable. We then say that the order of convergence is $n^{-\alpha}$, almost surely.

We show results of this type for a variety of problems. We begin by studying single server online routing problems. First, we introduce stochastic structure to city locations; under this framework, we show a capacity augmentation result (c.f. Chapter 4). Then, we add stochastic structure to the release dates and consider the online TSP with precedence constraints, the online TSP with capacity constraints and the online TRP. For these problems, we present algorithms that are almost surely asymptotically optimal. We then consider online machine scheduling problems. Introducing stochastic structure to release dates and processing times, we show that well-known online heuristics for a variety of parallel machine problems are almost surely asymptotically optimal.

Outline: In Section 5.2 we review the different notions of stochastic convergence. In Section 5.3, we introduce the stochastic assumptions and models we utilize for the routing problems we study. In Section 5.4 we present a capacity augmentation result under limited stochastic assumptions. In Section 5.5 we study the online TSP with precedence constraints and show that there exist algorithms that are almost surely asymptotically optimal. We show similar results in Section 5.6 for the online TSP with capacity constraints and Section 5.7 for the online TRP with precedence constraints. In Section 5.8, we introduce the models and stochastic assumptions we utilize in our asymptotic study of online machine scheduling problems. In Section 5.9 we study single machine scheduling problems and in Section 5.10 we study parallel machine scheduling problems.

5.2 Deterministic and Stochastic Convergence

In this section we review deterministic convergence as well as the different notions of stochastic convergence (i.e., convergence of random variables).

[1]We also utilize other modes of stochastic convergence, such as convergence in mean.

5.2.1 Deterministic Convergence

Let $x_1, x_2, \ldots$ denote a sequence of real numbers. We say that $\lim_{n\to\infty} x_n = c$ if, for any $\epsilon > 0$, there exists n_ϵ such that for all $n \geq n_\epsilon$

$$|x_n - c| \leq \epsilon.$$

We use $x_n \to c$ as shorthand for $\lim_{n\to\infty} x_n = c$.

5.2.2 Stochastic Convergence

We now consider the case where our sequence is random. Let $X_1, X_2, \ldots$ denote a sequence of random variables and let F_{X_i} denote the CDF of X_i. We are interested in the notion of *stochastic convergence*:

$$\lim_{n\to\infty} X_n = C, \tag{5.1}$$

where C could also be a random variable with distribution F_C. In this section, we summarize the notions that make Equation (5.1) precise. For more details, see Grimmett and Stirzaker [37].

Convergence in Distribution

We say that $X_n \to C$ *in distribution* if, for all t,

$$\lim_{n\to\infty} F_{X_n}(t) = F_C(t).$$

Note that the latter limit is (point-wise) deterministic. The Central Limit Theorem is an example of convergence in distribution.

Convergence in Probability

We say that $X_n \to C$ *in probability* if, for any $\epsilon > 0$,

$$\lim_{n\to\infty} \mathbb{P}[|X_n - C| \geq \epsilon] = 0.$$

Note that the latter limit is deterministic. The Weak Law of Large Numbers is an example of convergence in probability.

Convergence in probability implies convergence in distribution; i.e., if $X_n \to C$ in probability, then $X_n \to C$ in distribution.

Almost Sure Convergence

We say that $X_n \to C$ *almost surely* (a.s.) if

$$\mathbb{P}\left[\lim_{n\to\infty} X_n = C\right] = 1.$$

The Strong Law of Large Numbers is an example of almost sure convergence. Almost sure convergence implies convergence in probability and, hence, convergence in distribution.

Convergence in Mean

We say that $X_n \to C$ *in mean* if

$$\lim_{n\to\infty} \mathbb{E}\left[|X_n - C|\right] = 0.$$

Note that the latter limit is deterministic. The Strong Law of Large Numbers is also an example of convergence in mean. Convergence in mean implies convergence in probability and, hence, convergence in distribution.

5.3 Online Routing Preliminaries

In this section, we first define the stochastic routing models that we utilize in this chapter [2]. Note that we only study single server problems in this chapter. We then discuss the problem objectives and give a few classic limit results that we utilize in our analysis. Finally, we give a short discussion on our general technique for proving asymptotic optimality as well as comments on the choice of online algorithms.

5.3.1 City Locations

Let us briefly review the setting of our online routing problems. In an online routing problem, an online server must dynamically design paths or tours though a set of cities while respecting any constraints and minimizing a given objective. Unless otherwise stated, we assume the online and offline servers can move at a maximum of unit speed. Cities' existence and locations are revealed through a series of requests, which occur at time epochs denoted release dates (more on release dates in the next subsection). A request defines a set of cities and a precedence order in which the cities must be visited. Let there be n requests, each of which is revealed at its release date; furthermore, n is not known to the online server.

Let each request consist of m cities, which must be served in a given order. Comparing with the framework described in Chapter 2, Section 2.3, we let $k(i) = m$ for all i. Unless otherwise stated, we assume capacities are infinite in this chapter. Consider request k which consists of m cities $L_k^1, L_k^2, \ldots, L_k^m$. The cities must be served in the given order; i.e., location L_k^i must be visited before L_k^j for $i < j$. These requirements are denoted the *precedence constraints*. There are no precedence constraints across different requests, only within a single request. When $m = 1$ we say the request is *simple*; this situation will correspond to either the online TSP or TRP, depending on the objective. We now make some stochastic assumptions.

Assumption 3 *For each $i \in \{1, \ldots, m\}$, $L_1^i, L_2^i, \ldots, L_n^i$ must be independently iden-*

[2] Let us emphasize again that while the problem data is stochastic, the online algorithm does not have any information about the models nor the distributions.

tically distributed from a distribution of compact support in $d \geq 2$ dimensional Euclidean space. Additionally, L_k^i and L_l^j must be independent for all i, j, k, l (except, of course, when $i = j$ and $k = l$).

Remark 2 *Note that the distribution for $L_1^i, L_2^i, \ldots, L_n^i$ needs not be the same as the distribution for $L_1^j, L_2^j, \ldots, L_n^j$ for $i \neq j$. The support for the individual distributions do not even need to overlap.*

Let L_{TSP}^i denote the shortest tour through the locations $\{L_1^i, \ldots, L_n^i\}$ for each $i \in \{1, \ldots, m\}$. Finally, we define $d(L_1, L_2)$ to be the Euclidean distance between locations L_1 and L_2 and let o denote the origin.

5.3.2 City Release Dates

We now discuss the release dates of the requests. We let R_k denote the time that the k-th request is revealed; thus, $R_k \leq R_l$ for $k < l$. At time R_n the problem instance is completely revealed. We introduce two natural probabilistic structures for the release dates. We first consider a structure that is motivated by the uniformity of the requests and the second structure is motivated by the common use of the Poisson process in modeling arrivals over time.

Order Statistics

We take a somewhat myopic approach in defining our first release date structure. We assume that the release date of each request is a realization of a generic non-negative random variable $Y \geq 0$; i.e., the *unordered* release dates are independently identically distributed from a given distribution. As our model requires an order ($R_k \leq R_l$ for $k < l$), the k-th release date is the k-th order statistic:

$$R_k = Y_{(k)},$$

where $Y_k \geq 0$, $k = 1, \ldots, n$ are i.i.d. random variables and $Y_{(1)} \leq Y_{(2)} \leq \cdots, \leq Y_{(n)}$. Let μ_Y and σ_Y^2 denote the mean and variance, respectively, of the random variable

Y.

Renewal Processes

For our second structure, we consider a generic renewal process which is motivated by the usefulness of the Poisson process in modeling arrivals over time. Define non-negative i.i.d. random variables $X_i \geq 0$ to be the time between the $(i-1)^{th}$ and i^{th} release date. We then define the release dates as follows:

$$R_k = \sum_{i=1}^{k} X_i;$$

note that $R_{k+1} = R_k + X_{k+1}$ for all k. Let μ_X and σ_X^2 denote the mean and variance, respectively, of the random variable X.

5.3.3 Objectives

The objectives that we minimize are makespan and latency, motivated by the TSP and TRP, respectively. The makespan is the time required to visit all locations, respecting precedence constraints, and return to the origin. This objective may be interpreted to be in the server's interest. The latency is defined as the weighted sum of completion times $\sum_{i=1}^{n} w_i C_i$ ($w_i \geq 0$, $\forall i$), where the completion time of request i is the first time its m elements have been visited. The latency may be interpreted to be in the requests' interest.

5.3.4 Useful Limit Results

Under Assumption 3, we have the following well-known result by Beardwood, Halton and Hammersley [12]. This theorem, as well as generalizations of it, will prove central to our analysis.

Theorem 31 ([12]) *For each $i \in \{1, \ldots, m\}$ there exists a constant $c_{d,f_i} > 0$ such that*

$$\lim_{n\to\infty} \frac{L_{TSP}^i}{n^{(d-1)/d}} = c_{d,f_i}$$

almost surely and in mean, where f_i is the absolutely continuous part of the distribution of $L_1^i, \ldots, L_n^i$ and $d \geq 2$ is the dimension.

Next, we present two results that are akin to the Central Limit Theorem, but are for maxima as opposed to sums. As defined above, let $Y_{(n)} = \max_{1 \leq i \leq n} Y_i$; here, we consider only the case where the support of the Y random variables is $[0, \infty)$. Let F_Y be the CDF of the Y_i random variables. These results were originally given by Gnedenko [32]; we give the version presented in Chapter 2 of Galambos [31].

Theorem 32 ([32]) *If there is a constant γ such that, for all x,*

$$\lim_{t \to \infty} \frac{1 - F_Y(tx)}{1 - F_Y(t)} = x^{-\gamma},$$

then there exists a real sequence $(b_n) > 0$ such that

$$\lim_{n \to \infty} \mathbb{P}\left\{ \frac{Y_{(n)}}{b_n} \leq x \right\} = \begin{cases} \exp(-x^{-\gamma}), & x > 0 \\ 0, & o.w. \end{cases}$$

The normalizing constants (b_n) can be chosen as $b_n = \inf\{x \mid 1 - F_Y(x) \leq \frac{1}{n}\}$.

Before presenting our next result, we must define $R(t) = (1 - F_Y(t))^{-1} \int_0^\infty (1 - F_Y(y))dy$.

Theorem 33 ([32]) *If $\mu_Y < \infty$ and, for all real x,*

$$\lim_{t \to \infty} \frac{1 - F_Y(t + xR(t))}{1 - F_Y(t)} = e^{-x},$$

then there exist sequences (a_n) and $(b_n) > 0$ such that

$$\lim_{n \to \infty} \mathbb{P}\left\{ \frac{Y_{(n)} - a_n}{b_n} \leq x \right\} = \exp(-e^{-x}).$$

The normalizing constants (a_n) and (b_n) can be chosen as $a_n = \inf\{x \mid 1 - F_Y(x) \leq \frac{1}{n}\}$ and $b_n = R(a_n)$.

It has been shown (see [31]) that these two limit results are the only two possible for the maximum of unbounded random variables.

5.3.5 General Technique for Proving Asymptotic Optimality

We now describe the general technique we use to prove our results. In most cases, the random variables Z_n^*, Z_n^A and $C_n^A \triangleq Z_n^A - Z_n^*$ are difficult to characterize. To prove asymptotic optimality, we then consider lower and upper bounds. We find random variables F_n and G_n such that $Z_n^* \geq F_n$ and $Z_n^A \leq F_n + G_n$, which also implies $C_n^A \leq G_n$. Consequently, $\frac{C_n^A}{Z_n^*} \leq \frac{G_n}{F_n}$ (whenever $F_n > 0$) and we prove that $\lim_{n\to\infty} \frac{G_n}{F_n} = 0$.

In our analysis, one of the random variables F_n and G_n will be associated with the release dates of the problem and the other will be associated with the city locations. The essence of our proofs is that a meaningful function of the release dates does not grow (as a function of problems size) at the same rate as a meaningful function of the city locations.

5.3.6 On the Choice of Online Algorithm

Finally, we discuss the choice of algorithm. While we believe these convergence results will hold for a number of intelligent online algorithms, we concentrate on generic greedy algorithms (also called RE-OPT strategies in many papers). As an example, we state the greedy strategy for the online TSP: At any given release date (a new city enters the instance), the server will re-optimize and follow the shortest path, from its current location, through all unserved points and finally terminating at the origin. In other words, at any release date, the greedy strategy assumes the instance is complete and re-optimizes. More details are provided later for specific greedy strategies.

The analysis for the greedy algorithms will not assume any computational limitations; i.e., we assume we can solve NP-hard problems optimally. However, whenever possible, we extend our results to polynomial-time online algorithms.

5.4 Almost Sure Capacity Augmentation

We return to the setup of Chapter 2, Section 2.3, with the addition that city locations satisfy Assumption 3 and the restriction that each request consists of a single city (i.e., no precedence constraints). In other words, each request consists of a city location, release date and demand; city locations are stochastic while the release dates and demands are arbitrary. We now let the online algorithm have a single server with a capacity of Q and a speed $\gamma \geq 1$. The offline algorithm has a single server of unit speed with capacity $q \leq Q$. Q and q are constants. We begin with a lemma.

Lemma 16 *If $m = 1$ and $0 < \mathbb{E}[d(o, L)] < \infty$, then $\lim_{n\to\infty} \frac{Z_n^{r=0}(Q)}{Z_n^{r=0}(q)} = \frac{q}{Q}$, almost surely.*

Proof Since Q and q are constants, we can apply the results of [38], which state that

$$\frac{Z_n^{r=0}(Q)}{n} \to \frac{2\mathbb{E}[d(o,L)]}{Q} \quad \text{and} \quad \frac{Z_n^{r=0}(q)}{n} \to \frac{2\mathbb{E}[d(o,L)]}{q},$$

almost surely. Taking the limit of the ratio gives the result. ■

Theorem 34 *If $m = 1$ and $0 < \mathbb{E}[d(o, L)] < \infty$, then the asymptotic competitive ratio of Algorithm PAH-G is at most $1 + \frac{\rho q}{\gamma Q} + \frac{\rho - 1}{\gamma}$, almost surely.*

Proof The proof is very similar to that of Theorem 25; we highlight only the differences.

Case (1): The time needed by PAH-G is at most $r_n + \rho Z_n^{r=0}(Q)/\gamma \leq Z_n^*(q) + \rho Z_n^{r=0}(Q)/\gamma$. By Lemma 16, this upper bound is asymptotically equal to $Z_n^*(q) + \frac{\rho q}{\gamma Q} Z_n^{r=0}(q) \leq (1 + \frac{\rho q}{\gamma Q}) Z_n^*(q)$, almost surely.

Case (2a): Applying Lemma 16, we have that the online cost is almost surely at most

$$\begin{aligned} r_n + \frac{d(o, l_n^*)}{\gamma} + \rho \frac{Z_n^{r=0}(Q)}{\gamma} &\leq Z_n^*(q) + \left(\frac{2\rho - \gamma + 1}{2\gamma}\right) Z_n^{r=0}(Q) \\ &\to Z_n^*(q) + \left(\frac{2\rho - \gamma + 1}{2\gamma}\right)\left(\frac{q}{Q}\right) Z_n^{r=0}(q) \\ &\leq \left(1 + \left(\frac{2\rho - \gamma + 1}{2\gamma}\right)\left(\frac{q}{Q}\right)\right) Z_n^*(q) \end{aligned}$$

Case (2b): Since $Z^*(Q) \leq Z^*(q)$, the online cost is almost surely at most

$$\begin{aligned} Z^*(Q) + \frac{\rho}{\gamma} Z^{r=0}(Q) + \left(\frac{\rho-1}{\gamma}\right) Z^*(Q) &\leq Z^*(q) + \frac{\rho}{\gamma} Z^{r=0}(Q) + \left(\frac{\rho-1}{\gamma}\right) Z^*(q) \\ &\to Z^*(q) + \frac{\rho q}{\gamma Q} Z^{r=0}(q) + \left(\frac{\rho-1}{\gamma}\right) Z^*(q) \\ &\leq \left(1 + \frac{\rho q}{\gamma Q} + \frac{\rho-1}{\gamma}\right) Z^*(q). \end{aligned}$$

Since $\rho, \gamma \geq 1$ and $Q \geq q \geq 0$, $\max\left\{1 + \frac{\rho q}{\gamma Q}, 1 + \frac{\rho q}{\gamma Q} + \frac{\rho-1}{\gamma}, 1 + \left(\frac{2\rho-\gamma+1}{2\gamma}\right)\left(\frac{q}{Q}\right)\right\} = 1 + \frac{\rho q}{\gamma Q} + \frac{\rho-1}{\gamma}$ and the theorem is proved. ■

Corollary 17 *If we only have capacity augmentation, the asymptotic competitive ratio of Algorithm PAH-G is at most $\rho(1 + q/Q)$, almost surely.*

5.5 The Online TSP with Precedence Constraints

In this section, we again consider the setup of Chapter 2, Section 2.3 with some modifications. First, both online and offline servers have infinite capacities (which renders demands inconsequential). Next, we allow the data of the problem to be fully stochastic, as outlined in Section 5.3. Recall that the stochastic models of this chapter require each request to have the same number of cities ($k(i) = m$ for all i). Note that these models also require that the precedence constraints be an ordered list covering all cities in a request. When $m = 1$, we study the online Traveling Salesman Problem (TSP) and when $m = 2$ we study an online version of the makespan-objective Dial-a-Ride problem.

5.5.1 Algorithms and Bounds

We begin by defining a greedy strategy for the makespan objective, which we denote as Greedy-Makespan (GM).

Algorithm 16 : GM *At any release date, calculate a path $\mathcal{P}$ of shortest length that satisfies the following constraints:*

1. *$\mathcal{P}$ starts at the current server location and ends at the origin o.*

2. *All unserved requests are visited and the precedence constraints are respected.*

The server then traverses the path $\mathcal{P}$ at unit speed, until the next release date (if any).

We give two useful bounds for the online and offline costs.

Lemma 17

$$Z_n^* \geq \max\left\{R_n, \max_{1\leq i\leq m} L_{TSP}^i\right\}$$

and

$$Z_n^{GM} \leq \min\left\{R_n + \frac{3}{2}\sum_{i=1}^{m} L_{TSP}^i, 2R_n + \sum_{i=1}^{m} L_{TSP}^i\right\}.$$

Proof We first discuss the lower bounds on Z_n^*. Clearly, $Z_n^* \geq R_n$, the release date of the last request. Next, since for a given $i \in \{1,\ldots,m\}$ the locations $L_1^i,\ldots,L_n^i$ must be visited, the server must travel at least L_{TSP}^i, the shortest tour through these locations. As the server travels at unit speed,

$$Z_n^* \geq \max_{1\leq i\leq m} L_{TSP}^i.$$

We now consider strategy GM. At time R_n, the greedy server will optimize a path, from its current location through all remaining unserved points and finally returning to the origin. This greedy path will not take longer than the following alternate strategy: at time R_n, the server returns to the origin and then completes m tours in order. The first tour visits the locations $L_1^1,\ldots,L_n^1$ and takes L_{TSP}^1 units of time, since the server travels at unit speed. Then the server traverses L_{TSP}^2, followed by L_{TSP}^3 and so on, up to L_{TSP}^m. Clearly this is a feasible strategy. Thus strategy GM's cost may be bounded:

$$Z_n^{GM} \leq R_n + x + \sum_{i=1}^{m} L_{TSP}^i, \tag{5.2}$$

where x is the time required for the inital return to origin at time R_n. Since the server travels at unit speed, clearly $x \leq R_n$. Finally, the GM strategy will never allow the

server to proceed past the maximum city location; this gives

$$x \leq \max_{1 \leq i \leq m} \left\{ \max_{1 \leq j \leq n} \{d(o, L_j^i)\} \right\} \leq \max_{1 \leq i \leq m} \left\{ \frac{1}{2} L_{TSP}^i \right\} \leq \frac{1}{2} \sum_{i=1}^{m} L_{TSP}^i,$$

which gives the result. ■

Note that GM is not a polynomial-time strategy, since for even $m = 1$ we must calculate a Hamiltonian path. We now give a polynomial-time algorithm Greedy-Makespan-Polynomial (GMP) that gives a result almost as strong as Lemma 17.

Algorithm 17 : GMP *At any release date* R_j

1. *Return to the origin.*
2. *For each* $i \in \{1, \ldots, m\}$ *calculate a tour using Christofides' heuristic [25] to visit any unserved locations in* $\{L_1^i, \ldots, L_j^i\}$.
3. *Traverse the tours in the order* $1, 2, \ldots, m$ *at unit speed, until the next release date (if any).*

We have a corollary for algorithm GMP, which is easily seen by replacing Equation (5.2) in the proof of Lemma 17 with $Z_n^{GMP} \leq R_n + x + \frac{3}{2} \sum_{i=1}^{m} L_{TSP}^i$.

Corollary 18

$$Z_n^{GMP} \leq \min \left\{ R_n + 2 \sum_{i=1}^{m} L_{TSP}^i, 2R_n + \frac{3}{2} \sum_{i=1}^{m} L_{TSP}^i \right\}.$$

Finally, we consider algorithm PAH-p for the online TSP (i.e., $m = 1$), which was presented in Chapter 2. Using an argument similar to that in the proof of Lemma 17, we attain the following result.

Lemma 18

$$Z_n^{PAH-p} \leq R_n + 2\rho L_{TSP}.$$

Next we study each of the release date structures. We consider only $m = 1$ under the order statistic structure since our techniques were not successful for $m > 1$. Fortunately, under the renewal process structure we were able to prove an asymptotic optimality result for any m. Furthermore, for this latter structure, we show asymptotic optimality results for both polynomial-time algorithms GMP and PAH-p.

5.5.2 Order Statistic Release Dates

We only consider $m = 1$. We begin by stating the main result of this subsection.

Theorem 35 *If $m = 1$ and $\sigma_Y^2 < \infty$, then*

$$\lim_{n\to\infty} \frac{Z_n^{GM}}{Z_n^*} = 1$$

almost surely and in mean.

To prove Theorem 35, we begin with a useful lemma concerning R_n.

Lemma 19 ([11]) *If $\mathbb{E}[Y^r] < \infty$, $r \in \mathbb{N}$, then $\lim_{n\to\infty} \frac{Y_{(n)}}{n^\delta} = 0$ almost surely and in mean, for any $\delta \geq \frac{1}{r}$.*

Proof See the Appendix, Section A.2. ■

Proof of Theorem 35 As outlined in the Introduction, we find random variables F_n and G_n such that $Z_n^* \geq F_n$ and $Z_n^{GM} \leq F_n + G_n$. We then prove that $\lim_{n\to\infty} \frac{G_n}{F_n} = 0$. Now, with $m = 1$, we define $F_n = L_{TSP}$ and $G_n = 2R_n$, in accordance with Lemma 17. We prove the result for almost sure convergence; the proof for convergence in mean is identical. By Theorem 31, we have that there exists a $c_d > 0$ such that $\lim_{n\to\infty} \frac{F_n}{n^{(d-1)/d}} = c_d$ almost surely, and consequently, $\lim_{n\to\infty} \frac{n^{(d-1)/d}}{F_n} = \frac{1}{c_d}$ almost surely. By Lemma 19 and the fact that $\sigma_Y^2 < \infty$, we have that $\lim_{n\to\infty} \frac{G_n}{n^\delta} = 0$ almost surely for any $\delta \geq \frac{1}{2}$; we let $\delta = \frac{d-1}{d}$ for any $d \geq 2$. Multiplying the two latter limit results, we attain $\lim_{n\to\infty} \frac{G_n}{F_n} = 0$ almost surely, which proves the theorem. ■

Remark 3 *If $m > 1$ and we had choosen $F_n = \sum_{i=1}^{m} L_{TSP}^i$, it would have no longer been neccesarily true that $Z_n^* \geq F_n$ and our proof technique fails. Also, unfortunately, we were not able to attain a similar result for GMP (or PAH-p), even for $m = 1$, as choosing $F_n = \frac{3}{2} L_{TSP}$ does not necessarily satisfy $Z_n^* \geq F_n$.*

We now characterize the rates of convergence as described in the introduction to this chapter.

Bounded Y

We consider the situation where there exists a positive constant θ s.t. $\mathbb{P}\{Y \leq \theta\} = 1$ and consequently $\mathbb{P}\{R_i \leq \theta\} = 1$, for all i. By choosing the smallest such θ we can conclude that

$$\lim_{n \to \infty} R_n = \theta \ a.s.$$

Therefore, by choosing $\alpha = \frac{d-1}{d}$ and using Theorem 31, we have that

$$\lim_{n \to \infty} n^{(d-1)/d} \frac{R_n}{L_{TSP}} = \frac{\theta}{c_d},$$

a positive constant. We then say that the convergence is $n^{-(d-1)/d}$ almost surely.

Unbounded Y

We now consider the case of unbounded Y; i.e., the CDF $F_Y(y) < 1$ for all finite y. As the value of α will now depend on the distribution used, we consider two examples, each exemplifying the use of Theorems 32 and 33.

Example 1: Let Y have a Pareto distribution; i.e., $F_Y(y) = 1 - (\frac{\epsilon}{y})^\gamma$, $y \geq \epsilon > 0$. In our proof of convergence, we required a finite second moment of the random variable Y, which implies that $\gamma > 2$. Applying Theorem 32, we have that

$$\lim_{t \to \infty} \frac{1 - F_Y(ty)}{1 - F_Y(t)} = y^{-\gamma}.$$

Next, we calculate the normalizing constants,

$$b_n = \inf\{y \mid 1 - F_Y(y) \leq \frac{1}{n}\} = \inf\{y \mid \left(\frac{\epsilon}{y}\right)^{\gamma} \leq \frac{1}{n}\} = \epsilon n^{1/\gamma}.$$

Consequently, $\frac{R_n}{n^{1/\gamma}}$ converges in distribution to a non-degenerate random variable Z. Thus, by multipying $\frac{G_n}{F_n}$ by n^{α}, where $\alpha = \frac{d-1}{d} - \frac{1}{\gamma} > 0$, we get

$$\lim_{n\to\infty} n^{\alpha} \frac{R_n}{L_{TSP}} = \frac{Z}{c_d},$$

a non-degenerate random variable. We say that the convergence is $n^{-(\frac{d-1}{d} - \frac{1}{\gamma})}$, in distribution.

Example 2 We apply Theorem 33 with another example. Let Y be an Exponential random variable with unit parameter; i.e., $F_Y(y) = 1 - e^{-y}$. We first calculate $R(t) = 1$. Next, we see that

$$\lim_{t\to\infty} \frac{1 - F_Y(t + yR(t))}{1 - F_Y(t)} = \lim_{t\to\infty} \frac{e^{-(t+y)}}{e^{-t}} = e^{-y},$$

so the condition of the theorem is satisfied. We now calculate the normalizing constants:

$$a_n = \inf\{y \mid 1 - F_Y(y) \leq \frac{1}{n}\} = \inf\{y \mid e^{-y} \leq \frac{1}{n}\} = \ln(n) \quad \text{and} \quad b_n = R(a_n) = 1.$$

Consequently, $R_n - \ln(n)$ converges in distribution to a non-degenerate random variable Z. Thus, by *shifting* the random variable $\frac{R_n}{L_{TSP}}$ by $\frac{\ln(n)}{L_{TSP}}$ and then muitiplying by n^{α} with $\alpha = \frac{d-1}{d}$, we get

$$\lim_{n\to\infty} n^{\alpha} \left(\frac{R_n - \ln(n)}{L_{TSP}}\right) = \frac{Z}{c_d},$$

a non-degenerate random variable. We then say that the convergence is $n^{-(d-1)/d}$, in distribution.

5.5.3 Renewal Process Release Dates

We begin by stating and proving the main result of this subsection:

Theorem 36 *If* $0 < \mu_X < \infty$, *then*

$$\lim_{n\to\infty} \frac{Z_n^{GM}}{Z_n^*} = 1$$

almost surely and in mean. Furthermore, the rate of convergence is $n^{-1/d}$, *almost surely and in mean, respectively.*

Proof To prove this result, we first find appropriate random variables F_n and G_n: We assign $F_n = R_n$ and $G_n = \frac{3}{2}\sum_{i=1}^{m} L_{TSP}^i$, in accordance with Lemma 17. Again, we prove the theorem for almost sure convergence; the proof for convergence in mean is identical.

By Theorem 31, we have that, in dimension d, there exist positive constants $c_d^i > 0$ such that $\lim_{n\to\infty} \frac{L_{TSP}^i}{n^{(d-1)/d}} = c_d^i$ almost surely. Thus,

$$\lim_{n\to\infty} \frac{G_n}{n^{(d-1)/d}} = c_d, \ a.s., \tag{5.3}$$

where $c_d = \frac{3}{2}\sum_{i=1}^{m} c_d^i$. By the Strong Law of Large Numbers, we have that $\lim_{n\to\infty} \frac{F_n}{n} = \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} X_i = \mu_X$ almost surely. Since $\mu_X > 0$, we see that

$$\lim_{n\to\infty} \frac{n}{F_n} = \frac{1}{\mu_X}, \ a.s. \tag{5.4}$$

and, multiplying Equations (5.3) and (5.4), we see that $\lim_{n\to\infty} \frac{n^{1/d}G_n}{F_n} = \frac{c_d}{\mu_X}$ almost surely (this proves the rate of convergence). Finally, since $\lim_{n\to\infty} \frac{1}{n^{1/d}} = 0$, we conclude that $\lim_{n\to\infty} \frac{G_n}{F_n} = 0$ almost surely. ■

By substituting $G_n = 2\sum_{i=1}^{m} L_{TSP}^i$ in the proof of Theorem 36, we have the following corollary for the polynomial-time algorithm GMP.

Corollary 19 *If* $0 < \mu_X < \infty$, *then*

$$\lim_{n\to\infty} \frac{Z_n^{GMP}}{Z_n^*} = 1$$

almost surely and in mean. Furthermore, the rate of convergence is $n^{-1/d}$, *almost surely and in mean, respectively.*

Finally, by substituting $G_n = 2\rho L_{TSP}$ in the proof of Theorem 36, we have the following corollary for the polynomial-time algorithm PAH-p (for the online TSP, where $m = 1$).

Corollary 20 *If* $0 < \mu_X < \infty$, *then*

$$\lim_{n\to\infty} \frac{Z_n^{PAH-p}}{Z_n^*} = 1$$

almost surely and in mean. Furthermore, the rate of convergence is $n^{-1/d}$, *almost surely and in mean, respectively.*

5.6 The Online TSP with Capacity Constraints

In this section, we again consider the setup of Chapter 2, Section 2.3 with some modifications. First, each request consists of a single city: $k(i) = m = 1$ for all i; in effect, we eliminate the precedence constraints. Next, we allow the data of the problem to be stochastic, as outlined in Section 5.3. Request demands d_i may or may not be stochastic. Both online and offline servers have a capacity Q. Stated simply, we consider the online problem where a single server of capacity Q must serve simple requests, whose city locations and release dates are stochastic. We first study the case where the demands d_i are arbitrary and then we let the demands be stochastic as well.

5.6.1 Algorithms and Bounds

We define an online algorithm for this problem: Greedy-Capacitated-Routing (GCR).

Algorithm 18 : GCR *Whenever a new request is released, immediately return to the origin, calculate an optimal set of tours to visit the remaining unserved requests and begin traversing the tours (in arbitrary order) at unit speed.*

Lemma 20

$$Z_n^*(Q) \geq \max\left\{R_n, Z_n^{r=0}(Q)\right\}$$

and

$$Z_n^{GCR}(Q) \leq \min\left\{2R_n + Z_n^{r=0}(Q), R_n + \frac{1}{2}L_{TSP} + Z_n^{r=0}(Q)\right\}.$$

Proof Similar to the proof of Lemma 17. ■

Next, we consider both release date structures and give conditions under which we can show almost sure asymptotic optimality of GCR. We limit our discussion to requests being located in the two dimensional Euclidean plane; otherwise, Assumption 3 still holds, with $m = 1$.

5.6.2 Order Statistic Release Dates

We have two main results for this subsection. The first result is

Theorem 37

- *If* $d_i = 1,\ \forall i,\ Q$ *is constant,* $\mu_Y < \infty$ *and* $\mathbb{E}[d(o, L)] > 0$, *then*

$$\lim_{n\to\infty} \frac{Z_n^{GCR}(Q)}{Z_n^*(Q)} = 1$$

  almost surely.

- *If* $d_i = 1,\ \forall i,\ \lim_{n\to\infty} \frac{Q}{\sqrt{n}} = \infty$ *and* $\sigma_Y^2 < \infty$, *then*

$$\lim_{n\to\infty} \frac{Z_n^{GCR}(Q)}{Z_n^*(Q)} = 1$$

  almost surely.

This theorem is proved in part by using the following result given in [38].

Theorem 38 ([38])

- *If* $d_i = 1, \ \forall i$ *and* Q *is constant, then*

$$\lim_{n\to\infty} \frac{Z_n^{r=0}(Q)}{n} = \frac{2\mathbb{E}[d(o,L)]}{Q}$$

almost surely.

- *If* $d_i = 1, \ \forall i$ *and* $\lim_{n\to\infty} \frac{Q}{\sqrt{n}} = \infty$, *then there exists a constant* $\alpha > 0$ *where*

$$\lim_{n\to\infty} \frac{Z_n^{r=0}(Q)}{\sqrt{n}} = \alpha$$

almost surely.

Proof of Theorem 37 We prove the first part of the theorem first. We take $F_n = Z_n^{r=0}(Q)$, $G_n = 2R_n$ and show that $\lim_{n\to\infty} \frac{G_n}{F_n} = 0$ almost surely. We first decompose the argument of the limit:

$$\frac{G_n}{F_n} = 2\left(\frac{n}{Z_n^{r=0}(Q)}\right)\left(\frac{Y_{(n)}}{n}\right).$$

Using the first part of Theorem 38, we conclude that $\frac{n}{Z_n^{r=0}(Q)} \to \frac{Q}{2\mathbb{E}[d(o,L)]}$ almost surely. Since $\mu_Y < \infty$, Lemma 19 shows that $\frac{Y_{(n)}}{n} \to 0$ almost surely. We now prove the second part of the theorem and use the same random variables for F_n and G_n. Again, we decompose the argument of the limit:

$$\frac{G_n}{F_n} = 2\left(\frac{\sqrt{n}}{Z_n^{r=0}(Q)}\right)\left(\frac{Y_{(n)}}{\sqrt{n}}\right).$$

Using the second part of Theorem 38, we conclude that $\frac{\sqrt{n}}{Z_n^{r=0}(Q)} \to \frac{1}{\alpha}$ almost surely. Since $\sigma_Y^2 < \infty$, Lemma 19 shows that $\frac{Y_{(n)}}{\sqrt{n}} \to 0$ almost surely. ■

We now discuss the rates of convergence. If the Y random variable is bounded, then we can say the order of convergence is n^{-1} and $n^{-1/2}$ almost surely, for the first

and second parts of Theorem 37, respectively. We consider two examples (previously mentioned) to illustrate the convergence for unbounded Y. If Y is a Pareto random variable of parameter γ, then we can see that the convergence rates are $n^{-(1-\frac{1}{\gamma})}$ and $n^{-(\frac{1}{2}-\frac{1}{\gamma})}$, in distribution, for the first and second parts of the theorem, respectively. If Y is Exponential, of arbitrary parameter, we see that the convergence is n^{-1} and $n^{-1/2}$, in distribution, respectively.

Next, we consider a more general version of the online TSP with capacity constraints. We begin by normalizing all capacities and demands so that $Q = 1$ and $d_i \leq 1$, $\forall i$. We now allow the demands d_i to be random variables, i.i.d. from a distribution on $[0, 1]$. Under these conditions, the following result was proved by Bramel, Coffman, Shor and Simchi-Levi [19] and it will prove useful in showing another asymptotic optimality result.

Theorem 39 ([19]) *There exists a constant $\phi > 0$ such that*

$$\lim_{n\to\infty} \frac{Z_n^{r=0}(Q)}{n} = 2\phi\mathbb{E}[d(o, L)]$$

almost surely.

The second result of this subsection is the following.

Theorem 40 *If $\mu_Y < \infty$ and $\mathbb{E}[d(o, L)] > 0$, then*

$$\lim_{n\to\infty} \frac{Z_n^{GCR}(Q)}{Z_n^*(Q)} = 1$$

almost surely.

Proof We take $F_n = Z_n^{r=0}(Q)$, $G_n = 2R_n$ and prove that $\lim_{n\to\infty} \frac{G_n}{F_n} = 0$ almost surely. We rewrite $\frac{G_n}{F_n} = 2\left(\frac{n}{Z_n^{r=0}(Q)}\right)\left(\frac{Y_{(n)}}{n}\right)$. Using Theorem 39, we conclude that $\frac{n}{Z_n^{r=0}(Q)} \to (2\phi\mathbb{E}[d(o, L)])^{-1}$ almost surely. Since $\mu_Y < \infty$, Lemma 19 tells us that $\frac{Y_{(n)}}{n} \to 0$ almost surely. ■

If Y is bounded, the convergence rate is n^{-1} almost surely. Otherwise, we again consider the Pareto and Exponential distributions. Under the Pareto distribution, the

convergence rate is $n^{-(1-\frac{1}{\gamma})}$, in distribution and under the Exponential distribution, the convergence rate is n^{-1}, in distribution.

5.6.3 Renewal Process Release Dates

Our main result for this subsection is the following.

Theorem 41 *If* $d_i = 1$, $\forall i$, $\lim_{n\to\infty} \frac{Q}{\sqrt{n}} = \infty$ *and* $0 < \mu_X < \infty$, *then*

$$\lim_{n\to\infty} \frac{Z_n^{GCR}(Q)}{Z_n^*(Q)} = 1$$

almost surely. Furthermore, the rate of convergence is $n^{-1/2}$, *almost surely.*

Proof We take $F_n = R_n$, $G_n = \frac{1}{2}L_{TSP} + Z_n^{r=0}(Q)$ and prove that $\lim_{n\to\infty} \frac{G_n}{F_n} = 0$ almost surely. We first decompose the argument of the limit:

$$\frac{G_n}{F_n} = \left(\frac{n}{\sum_{i=1}^n X_i}\right)\left(\frac{1}{2}\frac{L_{TSP}}{\sqrt{n}} + \frac{Z_n^{r=0}(Q)}{\sqrt{n}}\right)\left(\frac{1}{\sqrt{n}}\right).$$

By the Strong Law of Large Numbers, we have that $\frac{\sum_{i=1}^n X_i}{n} \to \mu_X$ almost surely and since $\mu_X > 0$, $\frac{n}{\sum_{i=1}^n X_i} \to \frac{1}{\mu_X}$ almost surely. By Theorem 31, we have that there exists $c > 0$ such that $\frac{L_{TSP}}{\sqrt{n}} \to c$ almost surely. By Theorem 38, we have that $\frac{Z_n^{r=0}(Q)}{\sqrt{n}} \to \alpha$ almost surely. Finally, since $\frac{c+2\alpha}{2\mu_X}$ is a finite constant and $\frac{1}{\sqrt{n}} \to 0$, the theorem is proved. ■

5.7 The Online TRP with Precedence Constraints

In this section, we consider the setup of Chapter 2, Section 2.5 with some modifications. Each single city request is generalized to be a multiple-city request according to the model described in Section 5.3. We allow the data of the problem to be fully stochastic, as also outlined in Section 5.3. Recall that the stochastic models of this chapter require each request to have the same number of cities. Note that these models also require that the precedence constraints be an ordered list covering all cities in a request.

We study online routing problems where the objective is to minimize the latency, i.e., the sum of weighted completion times $\sum_{i=1}^{n} w_i C_i$, where $w_i \geq 0, \ \forall i$. The completion time of a request is the first time that all its m elements has been visited (of course, respecting the precedence constraints). Note that when $m = 1$ we have the online Traveling Repairman Problem and when $m = 2$, we have an online version of the latency-objective Dial-a-Ride problem. We now make the following assumption that simplifies the subsequent analysis.

Assumption 4 *There exist values* $0 < \omega \leq \Omega$ *such that* $\omega \leq w_i \leq \Omega, \ \forall i$.

The lower bound of ω in Assumption 4 simply eliminates cities with zero weight, cities which would not have been counted in the objective function cost anyway. The upper bound of Ω is intended to eliminate the pathological case where a single city has an arbitrary large weight which dominates the objective function cost.

5.7.1 Algorithms and Bounds

We define the strategy Greedy-Latency (GL) for these problems, followed by two polynomial-time strategies.

Algorithm 19 : GL *At any release date, calculate a path* $\mathcal{P}$ *of minimum latency that satisfies the following constraints:*

1. $\mathcal{P}$ *starts at the current server location.*
2. *All unserved requests are visited and the precedence constraints are respected.*
3. *If there are no unserved requests, remain idle at the current location (not necessarily the origin).*

The server then traverses the path $\mathcal{P}$ *at unit speed, until the next release date (if any).*

We next define the polynomial-time strategy Greedy-Latency-Polynomial (GLP) for the special case where $m = 1$ and $w_i = 1, \forall i$.

Algorithm 20 : GLP *At any release date, use a ρ-approximation algorithm for minimizing latency to find a path $\mathcal{P}$ beginning at the current server location and visiting all unserved requests. Then the server traverses $\mathcal{P}$ at unit speed, until the next release date (if any). If there are no unserved locations, remain idle at the current location (not necessarily the origin).*

Remark 4 *To the best of our knowledge, there are no approximation algorithms for the arbitrary weight case. Also to the best of our knowledge, the approximation algorithm (for the unit weight case) with the smallest ratio ρ to date is the one given by Chaudhuri, Godfrey, Rao and Talwar [21], which has $\rho < 3.6$.*

Finally, we give a simple polynomial-time algorithm for the general case: Serve-In-Order-Received (SIOR).

Algorithm 21 : SIOR *Serve in the order received; i.e., visit the locations in the order:*

$$L_1^1, \ldots, L_1^m, L_2^1, \ldots, L_2^m, \ldots, L_n^1, \ldots, L_n^m.$$

When there are no known unserved locations, remain idle at the current location.

We now derive useful bounds for the costs of these algorithms, as well as for the optimal offline cost, in a series of lemmas and corollaries. We consider separately the cases $m = 1$ and an arbitrary value of m.

We first consider the case where $m = 1$. We need an additional definition for the statement of the first lemma. Let $L_{TRP} = \sum_{i=1}^{n} w_i C_i^*$ be the optimal value of the latency for the locations $L_1, \ldots, L_n$, when all release dates are zero.

Lemma 21 *If $m = 1$, $Z_n^* \geq L_{TRP}$ and $Z_n^{GL} \leq 2R_n \sum_{i=1}^{n} w_i + L_{TRP}$.*

Proof The lower bound on Z_n^* is clear. Now we consider the server (repairman) at time R_n. Consider an alternate strategy where the server returns to the origin and then servers *all* cities optimally; this strategy clearly has a larger latency than GL since GL does not necessarily return to the origin at time R_n and may have already

served some cities. The initial return to the origin of this alternate strategy takes at most R_n time since the server moves at unit speed. The (alternate) server then proceeds on the optimal path that minimizes the latency through all n cities. The completion time of request i in the alternate strategy is $2R_n + C_i^*$, which implies that the cost of GL is at most $2R_n \sum_{i=1}^{n} w_i + L_{TRP}$. ■

The following corollary is immediate.

Corollary 21 *If $m = 1$ and $w_i = 1$, $\forall i$, $Z_n^{GLP} \leq 2nR_n + \rho L_{TRP}$.*

We now consider the situation where m is arbitrary.

Lemma 22

$$Z_n^* \geq \sum_{j=1}^{n} w_j R_j \quad \textit{and} \quad Z_n^{GL} \leq \sum_{j=1}^{n} w_j \left(R_j + 3 \sum_{i=1}^{m} L_{TSP}^i \right).$$

Proof We begin with the lower bound on Z_n^*. Clearly, the optimal completion time of each request is at least its release date; thus we have $Z_n^* \geq \sum_{j=1}^{n} w_j R_j$.

We now show the upper bound on Z_n^{GL} by induction on the number of requests n. For $n = 1$ (subscripts are supressed), with $L^0 = o$, it is clear that

$$\begin{aligned} Z^{GL} &= w \left(R + \sum_{i=1}^{m} d(L^{i-1}, L^i) \right) \\ &\leq w \left(R + \sum_{i=1}^{m} \left(d(L^{i-1}, o) + d(o, L^i) \right) \right) \\ &\leq w \left(R + \sum_{i=1}^{m} 2d(o, L^i) \right) \\ &= w \left(R + \sum_{i=1}^{m} L_{TSP}^i \right). \end{aligned}$$

Now, assuming $Z_{n-1}^{GL} \leq \sum_{j=1}^{n-1} w_j \left(R_j + 3 \sum_{i=i}^{m} L_{TSP}^i(n-1) \right)$, $L_{TSP}^i(n-1)$ being the shortest tour through the locations $L_1^i, \ldots, L_{n-1}^i$, and noting that $L_{TSP}^i(n-1) \leq$

$L^i_{TSP}(n) \triangleq L^i_{TSP}$, we shall prove the result for n. Define C^{max}_{n-1} as the (projected) maximum completion time of all requests in the instance of $(n-1)$ requests. We first find an upper bound on C^{max}_{n-1}. Recall that GL performed a re-optimization at time R_{n-1}. Consider an alternate server that, at time R_{n-1}, first returned to the origin before proceeding to visit all unserved requests; this return takes at most

$$\max_{1\le i\le m}\{\max_{1\le j\le n-1} d(o, L^i_j)\} \le \max_{1\le i\le m}\{\frac{1}{2}L^i_{TSP}\} \le \frac{1}{2}\sum_{i=1}^{m} L^i_{TSP}$$

time. Once the alternate server reaches the origin, it first travels through the locations $\{L^1_1, \ldots, L^1_{n-1}\}$, then $\{L^2_1, \ldots, L^2_{n-1}\}$ and so on until $\{L^m_1, \ldots, L^m_{n-1}\}$. This takes at most $\sum_{i=1}^{m} L^i_{TSP}$ time. Since C^{max}_{n-1} for GL is clearly at most the respective value for this alternate strategy, we have that

$$C^{max}_{n-1} \le R_{n-1} + \frac{3}{2}\sum_{i=1}^{m} L^i_{TSP} \le R_n + \frac{3}{2}\sum_{i=1}^{m} L^i_{TSP}.$$

Re-optimizing at time R_n will result in a latency value that is no more than that of the following strategy: Wait until requests $1, \ldots, (n-1)$ have all been served and then serve request n. Letting $\tilde{C}_n$ denote the completion time of request n in this virtual strategy and noting that at time C^{max}_{n-1} the server is at a location L^m_j, $j \in \{1, \ldots, n-1\}$, we have that

$$\begin{aligned}
Z^{GL}_n &\le Z^{GL}_{n-1} + w_n\tilde{C}_n \\
&= Z^{GL}_{n-1} + w_n\left(C^{max}_{n-1} + d(L^m_j, L^1_n) + \sum_{i=2}^{m} d(L^{i-1}_n, L^i_n)\right) \\
&\le Z^{GL}_{n-1} + w_n\left(C^{max}_{n-1} + d(L^m_j, o) + d(o, L^1_n) + \sum_{i=2}^{m}\left(d(L^{i-1}_n, o) + d(o, L^i_n)\right)\right) \\
&\le Z^{GL}_{n-1} + w_n\left(C^{max}_{n-1} + \frac{1}{2}L^m_{TSP} + \sum_{i=1}^{m} 2d(o, L^i_n)\right) \\
&\le Z^{GL}_{n-1} + w_n\left(C^{max}_{n-1} + \frac{1}{2}L^m_{TSP} + \sum_{i=1}^{m} L^i_{TSP}\right)
\end{aligned}$$

$$\leq Z_{n-1}^{GL} + w_n \left(C_{n-1}^{max} + \frac{3}{2} \sum_{i=1}^{m} L_{TSP}^i \right)$$
$$\leq Z_{n-1}^{GL} + w_n \left(R_n + 3 \sum_{i=1}^{m} L_{TSP}^i \right);$$

applying the inductive hypothesis proves the lemma. ■

The proof of Lemma 22 also directly applies to strategy *SIOR*:

Corollary 22

$$Z_n^{SIOR} \leq \sum_{j=1}^{n} w_j \left(R_j + 3 \sum_{i=1}^{m} L_{TSP}^i \right).$$

5.7.2 Order Statistic Release Dates

Our main result for this subsection is the following.

Theorem 42 *If* $m = 1$, $w_i = 1$, $\forall i$, $\mathbb{E}[Y^3] < \infty$ *and* $L_1, \ldots, L_n$ *are uniformly distributed in* $[0,1]^2$, *then*

$$\lim_{n \to \infty} \frac{Z_n^{GL}}{Z_n^*} = 1$$

almost surely.

In order to prove this result, we need the following result by Bompadre, Dror and Orlin [17].

Lemma 23 ([17]) *The cost* L_{TRP} *of the (offline) minimum latency problem (with unit weights) when* n *cities are uniformly distributed in* $[0,1]^2$ *is* $\Omega(n^{3/2})$ *almost surely.*

Proof of Theorem 42 We first find appropriate random variables F_n and G_n. By Lemma 21 we let $F_n = L_{TRP}$ and $G_n = 2nR_n$. By Corollary 23, we have that $L_{TRP} = \Omega(n^{3/2})$ almost surely. Since L_{TRP} is almost surely positive, we may conclude that $\frac{1}{L_{TRP}} = O(\frac{1}{n^{3/2}})$ almost surely. For any $\epsilon > 0$, we have that $\frac{1}{L_{TRP}} = o(\frac{n^\epsilon}{n^{3/2}})$ almost

surely. Equivalently, we have that $\lim_{n\to\infty} \frac{n^\gamma}{L_{TRP}} = 0$ almost surely, for any $\gamma < \frac{3}{2}$. Next, we decompose the limit:

$$\frac{G_n}{F_n} = \frac{2nY_{(n)}}{L_{TRP}} = 2\frac{n^{4/3}}{L_{TRP}}\frac{Y_{(n)}}{n^{1/3}}.$$

Taking limits, with $\gamma = \frac{4}{3}$ and applying Lemma 19 (with $r = 3$ and $\delta = \frac{1}{3}$), proves the theorem. ■

Remark 5 *We actually only require that there exists $\varepsilon > 0$ such that $\mathbb{E}[Y^{2+\varepsilon}] < \infty$ to prove the above theorem.*

Remark 6 *Unfortunately, we were unable to prove a similar result for GLP. Corollary 21 and the proof of Theorem 42 suggest choosing $F_n = \rho L_{TRP}$. But since $\rho > 1$, it would have no longer been necessarily true that $Z_n^* \geq F_n$.*

We have not been able to say anything about the rate of convergence in this case, since the only information we know about the asymptotic behavior of L_{TRP}, the optimal classic TRP cost, is a lower bound of $\Omega(n^{3/2})$ almost surely.

5.7.3 Renewal Process Release Dates

After giving a useful lemma, we state and prove the main result of this subsection.

Lemma 24 ([11]) *If $\mathbb{E}[X^4] < \infty$, then for any $k \geq 0$,*

$$\lim_{n\to\infty} \frac{1^k X_1 + 2^k X_2 + 3^k X_3 + \cdots + n^k X_n}{n^{k+1}} = \frac{\mu_X}{k+1},$$

almost surely.

Proof See the Appendix, Section A.2. ■

Theorem 43 *If $\mu_X > 0$ and $\mathbb{E}[X^4] < \infty$, then*

$$\lim_{n\to\infty} \frac{Z_n^{GL}}{Z_n^*} = 1$$

almost surely. Furthermore, the rate of convergence is $n^{-1/d}$, almost surely.

Proof We assign $F_n = \sum_{j=1}^{n} w_j R_j$ and $G_n = 3\left(\sum_{j=1}^{n} w_j\right)\left(\sum_{i=1}^{m} L_{TSP}^i\right)$, in accordance with Lemma 22. Showing $\lim_{n\to\infty} \frac{G_n}{F_n} = 0$ almost surely proves the theorem. We first bound (using Assumption 4) the argument of the limit:

$$\frac{G_n}{F_n} = \frac{3\left(\sum_{j=1}^{n} w_j\right)\left(\sum_{i=1}^{m} L_{TSP}^i\right)}{\sum_{j=1}^{n} w_j R_j} \leq \frac{3n\Omega \sum_{i=1}^{m} L_{TSP}^i}{\omega \sum_{j=1}^{n} R_j}.$$

We now express the sum of release dates in terms of the X random variables:

$$\sum_{i=1}^{n} R_i = \sum_{i=1}^{n}\sum_{j=1}^{i} X_j = \sum_{j=1}^{n}\sum_{i=j}^{n} X_j = \sum_{j=1}^{n}(n-j+1)X_j = \sum_{j=1}^{n} jX_j,$$

where the last equality follows (almost surely) from the fact that the X_j random variables are i.i.d. Next, we take limits and apply Lemma 24 with $k = 1$ and Theorem 31:

$$\begin{aligned}
\frac{3n\Omega \sum_{i=1}^{m} L_{TSP}^i}{\omega \sum_{j=1}^{n} R_j} &= \frac{3n\Omega \sum_{i=1}^{m} L_{TSP}^i}{\omega \sum_{j=1}^{n} jX_j} \\
&= \left(\frac{3\Omega}{\omega}\right)\left(\frac{n^2}{\sum_{j=1} jX_j}\right)\left(\frac{\sum_{i=1}^{m} L_{TSP}^i}{n^{(d-1)/d}}\right)\left(\frac{1}{n^{1/d}}\right) \\
&\to \left(\frac{3\Omega}{\omega}\right)\left(\frac{2}{\mu_X}\right)(c_d)(0),
\end{aligned}$$

almost surely, and the convergence is proved. Finally, by noting that $n^{1/d}\frac{G_n}{F_n} \to \frac{6\Omega c_d}{\omega \mu_X} > 0$, we attain the desired rate of convergence. ■

Since the upper bound on the cost of SIOR is identical to that of GL (c.f. Lemma 22 and Corollary 22), we have the following corollary for the polynomial-time SIOR.

Corollary 23 *If $\mu_X > 0$ and $\mathbb{E}[X^4] < \infty$, then*

$$\lim_{n\to\infty} \frac{Z_n^{SIOR}}{Z_n^*} = 1$$

almost surely. Furthermore, the rate of convergence is $n^{-1/d}$, almost surely.

5.8 Online Machine Scheduling Preliminaries

In this section, we study the asymptotic properties of online algorithms for a number of machine scheduling problems whose offline versions are NP-hard. We are given a set of n jobs, each with a processing requirement $p_i \geq 0$, a weight $w_i \geq 0$ and a release date $r_i \geq 0$. We consider scheduling jobs on (1) a single machine, (2) parallel identical machines and (3) parallel uniform machines (i.e. machines have different speeds). The objective is to schedule the jobs on the machine(s) to minimize the weighted sum of completion dates $\sum_j w_j C_j$. We shall consider both preemptive and non-preemptive problems. In scheduling notation [36], we study online versions of $1|r_j, pmtn|\sum_j w_j C_j$, $1|r_j|\sum_j w_j C_j$, $Q|r_j, pmtn|\sum_j C_j$ and $P|r_j|\sum_j C_j$. Recall that P indicates parallel identical machines and Q indicates parallel uniform machines of different speeds. Let $s_1 \geq s_2 \geq \cdots s_m > 0$ be the speeds of the machines in the Q case; in the P case, $s_j = 1$, $\forall j$. Job i on machine j will take p_i/s_j time to complete.

We analyze a number of well-known algorithms from a new perspective. In particular, we make very general probabilistic assumptions on the problem data; no specific distributional assumptions are made. Under these assumptions, we show that these algorithms are almost surely asymptotically optimal. The main motivation for this research is to provide an explanation of the excellent performance some of these algorithms exhibit computationally as well as in practice. Additionally, our research suggests that untested online algorithms will perform well in practice.

5.8.1 Stochastic Assumptions

We introduce an underlying probabilistic structure to the problems, which is unknown and unused by the online algorithms, and use these probabilistic assumptions as a tool to study deterministic problems. Furthermore, we do not assign a specific distribution to the data, but rather a very general class of distributions.

We let the processing requirements p_i be modeled as i.i.d. non-negative random variables P_i with common distribution f_P. We let the weights be i.i.d. non-negative random variables W_i with common distribution f_W. Finally, we let the release dates

be i.i.d. non-negative random variables R_i with common distribution f_R. We also define $R_{(n)} = \max_{1 \le i \le n} R_i$ to be the release date of the final job and $P_{(n)} = \max_{1 \le i \le n} P_i$ the largest processing requirement. Note that no other assumptions about the distributions f_P, f_W and f_R are needed.

5.8.2 Technical Details

We mention a useful result.

Lemma 25 ([11]) *Let $\{X_i\}$ and $\{Y_j\}$ be two sequences of i.i.d. random variables. If $\mathbb{E}[X^2] < \infty$ and $\mathbb{E}[Y^2] < \infty$, then*

$$\lim_{n \to \infty} \frac{\sum_{j=1}^{n} X_j \sum_{i=1}^{j} Y_i}{n^2} = \frac{\mathbb{E}[X]\mathbb{E}[Y]}{2}, \ a.s.$$

Proof See the Appendix, Section A.2. ■

5.9 Single Machine Minsum Online Scheduling

In this section, we consider online versions of the single machine scheduling problems $1|r_j, pmtn| \sum_j w_j C_j$ and $1|r_j| \sum_j w_j C_j$; offline versions of both these problems are NP-hard.

5.9.1 Online $1|r_j, pmtn| \sum_j w_j C_j$

Consider the preemptive *Weighted Shortest Processing Requirement* (WSPR) heuristic, which is an online algorithm: At any point in time, among the known unfinished jobs, process the one with the highest ratio w_i/p_i. Note that the WSPR heuristic solves $1|| \sum_j w_j C_j$, and consequently $1|pmtn| \sum_j w_j C_j$, exactly; see Pinedo [59] . We begin by stating the main result for this subsection.

Theorem 44 *If* $\mathbb{E}[R] < \infty$, $\mathbb{E}[P] > 0$, $\mathbb{E}[W] > 0$, $\mathbb{E}[P^2] < \infty$, *and* $\mathbb{E}[W^2] < \infty$, *then the WSPR heuristic is almost surely asymptotically optimal for the online version of* $1|r_j, pmtn| \sum_j w_j C_j$.

Proof Let Z_n^{WSPR} be the random variable denoting the cost of WSPR on an instance of n jobs under the probabilistic conditions of the theorem. Let Z_n^* be the random variable denoting the optimal offline cost for $1|r_j, pmtn| \sum_j w_j C_j$. Finally, let $Z_n^{\mathcal{R}}$ be the random variable for the cost of the relaxed problem $1|pmtn| \sum_j w_j C_j$, which is solved optimally by the WSPR heuristic; clearly $Z_n^{\mathcal{R}} \leq Z_n^*$.

At time $R_{(n)}$, the release date of the final job in the instance, assume that no processing has been done; clearly, this will only increase the online cost of WSPR. Therefore, under this assumption, at time $R_{(n)}$, the WSPR heuristic essentially sees the problem $1|pmtn| \sum_j w_j C_j$ (i.e., all release dates are equal to zero). Consequently, we have that

$$Z_n^{WSPR} \leq R_{(n)} \sum_{j=1}^{n} W_j + Z_n^{\mathcal{R}}. \tag{5.5}$$

Considering the ratio of online to offline costs, we have that

$$\begin{aligned} \frac{Z_n^{WSPR}}{Z_n^*} &\leq \frac{R_{(n)} \sum_{j=1}^{n} W_j + Z_n^{\mathcal{R}}}{Z_n^{\mathcal{R}}} \\ &= 1 + \frac{R_{(n)} \sum_{j=1}^{n} W_j}{Z_n^{\mathcal{R}}}. \end{aligned}$$

Next, in order to compute $Z_n^{\mathcal{R}}$, we re-order the indices on the W and P random variables such that

$$\frac{W_1}{P_1} \geq \frac{W_2}{P_2} \geq \cdots \geq \frac{W_n}{P_n};$$

this operation is without loss of generality since the two sequences of random variables (P_i), and (W_i) are i.i.d. This re-indexing allows us to find a closed-form expression for $1|pmtn| \sum_j w_j C_j$. The completion time of the first job processed, job 1, is P_1; the completion time of job 2 is $P_1 + P_2$; the completion time of job k is $P_1 + \cdots + P_k$.

Therefore, the cost of $1|pmtn|\sum_j w_j C_j$ is

$$Z_n^{\mathcal{R}} = \sum_{j=1}^{n} W_j \sum_{i=1}^{j} P_i.$$

We again consider the ratio of the cost of WSPR to the optimal offline cost:

$$\frac{Z_n^{WSPR}}{Z_n^*} \leq 1 + \frac{R_{(n)} \sum_{j=1}^{n} W_j}{Z_n^{\mathcal{R}}} = 1 + \frac{R_{(n)} \sum_{j=1}^{n} W_j}{\sum_{j=1}^{n} W_j \sum_{i=1}^{j} P_i}. \tag{5.6}$$

Next, we decompose the fraction in the RHS of Equation (5.6), take limits as $n \to \infty$ and, using Lemma 19, the SLLN and Lemma 25, we have that:

$$\begin{aligned} \frac{R_{(n)} \sum_{j=1}^{n} W_j}{\sum_{j=1}^{n} W_j \sum_{i=1}^{j} P_i} &= \left(\frac{R_{(n)}}{n}\right)\left(\frac{\sum_{j=1}^{n} W_j}{n}\right)\left(\frac{n^2}{\sum_{j=1}^{n} W_j \sum_{i=1}^{j} P_i}\right) \\ &\to (0)\,(\mathbb{E}[W])\left(\frac{2}{\mathbb{E}[W]\mathbb{E}[P]}\right) \\ &= 0, \end{aligned}$$

almost surely. Therefore, $\lim_{n\to\infty} \frac{Z_n^{WSPR}}{Z_n^*} = 1$ almost surely, which means that WSPR is almost surely asymptotically optimal for the online version of $1|r_j, pmtn|\sum_j w_j C_j$. ■

5.9.2 Online $1|r_j|\sum_j w_j C_j$

A non-preemptive version *Non-preemptive Weighted Shortest Processing Requirement* (NWSPR) of WSPR is easily defined: Whenever the machine is available to process a job, if there remain unprocessed jobs, choose the job with the highest ratio w_i/p_i. We are able to prove the exact same result as Theorem 44:

Theorem 45 *If* $\mathbb{E}[R] < \infty$, $\mathbb{E}[P] > 0$, $\mathbb{E}[W] > 0$, $\mathbb{E}[P^2] < \infty$, *and* $\mathbb{E}[W^2] < \infty$, *then the NWSPR heuristic is almost surely asymptotically optimal for the online version of* $1|r_j|\sum_j w_j C_j$.

The proof of Theorem 45 is very similar to that for Theorem 44; we highlight only the *differences.*

Proof First, note that $Z_n^{\mathcal{R}}$ is the optimal solution for both $1|pmtn|\sum_j w_j C_j$ and $1||\sum_j w_j C_j$, since these two problems are essentially identical. Equation (5.5) is modified to become

$$Z_n^{NWSPR} \leq \left(R_{(n)} + P_{(n)}\right) \sum_{j=1}^{n} W_j + Z_n^{\mathcal{R}}.$$

The reason for this modification is because at time $R_{(n)}$, we can not relate NWSPR's actions to the problem $1||\sum_j w_j C_j$, since it might be busy processing some job. But after $P_{(n)} = \max_{1 \leq i \leq n} P_i$ time, we are certain that the machine has finished whatever job had been in progress at time $R_{(n)}$. Therefore, at time $R_{(n)} + P_{(n)}$, assuming that no job has been processed, NWSPR "sees" the problem $1||\sum_j w_j C_j$. The only other significant modification is the final decomposition:

$$\begin{aligned} \frac{\left(R_{(n)} + P_{(n)}\right) \sum_{j=1}^{n} W_j}{\sum_{j=1}^{n} W_j \sum_{i=1}^{j} P_i} &= \left(\frac{R_{(n)}}{n} + \frac{P_{(n)}}{n}\right) \left(\frac{\sum_{j=1}^{n} W_j}{n}\right) \left(\frac{n^2}{\sum_{j=1}^{n} W_j \sum_{i=1}^{j} P_i}\right) \\ &\to (0+0)\left(\mathbb{E}[W]\right) \left(\frac{2}{\mathbb{E}[W]\mathbb{E}[P]}\right) \\ &= 0, \end{aligned}$$

almost surely. The rest of the proof remains identical. ■

5.10 Parallel Machine Minsum Online Scheduling

We consider online versions of the parallel machine scheduling problems $Q|r_j, pmtn|\sum_j C_j$ and $P|r_j|\sum_j C_j$; offline versions of both these problems are NP-hard.

5.10.1 Online $Q|r_j, pmtn|\sum_j C_j$

Consider the *Shortest Remaining Processing Requirement on Fastest Machine* (SRPR-FM) heuristic, which is also an online algorithm: At any given time, the job with the shortest remaining processing requirement is assigned to the fastest machine, the job

with the second shortest remaining processing requirement is assigned to the second fastest machine, and so on. Note that the SRPR-FM heuristic solves $Q|pmtn|\sum_j C_j$ exactly; see [59].

The reason that we only consider unit weights is that even $P|pmtn|\sum_j w_j C_j$ is NP-hard and our technique for proving asymptotical optimality would break down, since we would require a closed form expression for the optimal cost of this NP-hard problem.

Theorem 46 *If* $\mathbb{E}[R] < \infty$, $\mathbb{E}[P] > 0$, *and* $\mathbb{E}[P^2] < \infty$, *then the SRPR-FM heuristic is almost surely asymptotically optimal for the online version of* $Q|r_j, pmtn|\sum_j C_j$.

Proof Let $Z_n^{SRPR-FM}$ be the random variable denoting the cost of SRPR-FM on an instance of n jobs under the probabilistic conditions of the theorem. Let Z_n^* be the random variable denoting the optimal offline cost for $Q|r_j, pmtn|\sum_j C_j$. Finally, let $Z_n^{\mathcal{R}}$ be the random variable for the cost of the relaxed problem $Q|pmtn|\sum_j C_j$, which is solved optimally by the SRPR-FM heuristic; clearly $Z_n^{\mathcal{R}} \leq Z_n^*$.

At time $R_{(n)}$, the release date of the final job in the instance, assume that no processing has been done; clearly, this will only increase the online cost of SRPR-FM. Therefore, under this assumption, at time $R_{(n)}$, the SRPR-FM heuristic essentially sees the problem $Q|pmtn|\sum_j C_j$. Consequently, we have that

$$Z_n^{SRPR-FM} \leq nR_{(n)} + Z_n^{\mathcal{R}}. \tag{5.7}$$

Considering the ratio of online to offline costs, we have that

$$\begin{aligned} \frac{Z_n^{SRPR-FM}}{Z_n^*} &\leq \frac{nR_{(n)} + Z_n^{\mathcal{R}}}{Z_n^{\mathcal{R}}} \\ &= 1 + \frac{nR_{(n)}}{Z_n^{\mathcal{R}}}. \end{aligned}$$

Next, we wish to compute a lower bound on $Z_n^{\mathcal{R}}$. Clearly, the optimal cost of $P|pmtn|\sum_j C_j$, when all machines have speed s_1 (the fastest machine), is a lower bound; we now analyze this latter problem. Note that the non-preemptive Shortest-

Processing-Requirement (SPR) [3] heuristic solves $P|pmtn|\sum_j C_j$ exactly; see [59]. Defining $N = \{1, \ldots, n\}$ as the set of all jobs, note that the SPR heuristic will induce a partition of N into $N_1, N_2, \ldots, N_m$, where N_i is the set of jobs that are completed on machine i; clearly $n = \sum_{i=1}^{m} |N_i|$. Therefore,

$$Z_n^{\mathcal{R}} \geq \sum_{j \in N} C_j^{SPR} = \sum_{i=1}^{m} \sum_{j \in N_i} C_j^{SPR}.$$

Now, consider N_i for any $i \in \{1, \ldots, m\}$. We re-name the indices for P_j, $j \in N_i$ such that

$$P_1 \leq P_2 \leq \cdots \leq P_{|N_i|}.$$

Considering machine i, the completion time of the first job processed, job 1, is P_1; the completion time of job 2 is $P_1 + P_2$; the completion time of job k is $P_1 + \cdots + P_k$. Therefore, the cost contribution of machine i is

$$\sum_{j \in N_i} C_j^{SPR} = \sum_{j=1}^{|N_i|} \sum_{k=1}^{j} P_k / s_1$$

and, consequently, the optimal cost of $P|pmtn|\sum_j C_j$ is

$$\sum_{i=1}^{m} \sum_{j=1}^{|N_i|} \sum_{k=1}^{j} P_k / s_1.$$

Now, let $x \in \mathbb{R}^m$ be defined as $x = (|N_1|, |N_2|, \ldots, |N_m|)$ and $e \in \mathbb{R}^m$ be defined as the vector of ones. By the Cauchy-Schwarz Inequality, $n = x'e \leq \|x\| \cdot \|e\|$ and, consequently, $n^2 \leq \|x\|^2 \cdot \|e\|^2 = (\sum_{i=1}^{m} |N_i|^2)(m)$. Thus,

$$1 \leq n^{-2} m \sum_{i=1}^{m} |N_i|^2. \tag{5.8}$$

[3] Whenever a machine is available to process a job, choose the job with the shortest processing requirement.

We again consider the ratio of the cost of SRPR-FM to the optimal offline cost:

$$\frac{Z_n^{SRPR-FM}}{Z_n^*} \leq 1 + \frac{nR_{(n)}}{Z_n^{\mathcal{R}}} \leq 1 + \frac{nR_{(n)}s_1}{\sum_{i=1}^{m}\sum_{j=1}^{|N_i|}\sum_{k=1}^{j} P_k}. \tag{5.9}$$

Next, using Equation (5.8), we bound and decompose the fraction in the RHS of Equation (5.9):

$$\begin{aligned}
\frac{nR_{(n)}s_1}{\sum_{i=1}^{m}\sum_{j=1}^{|N_i|}\sum_{k=1}^{j} P_k} &\leq \left(\frac{m\sum_{l=1}^{m}|N_l|^2}{n^2}\right)\left(\frac{nR_{(n)}}{\sum_{i=1}^{m}\sum_{j=1}^{|N_i|}\sum_{k=1}^{j} P_k}\right)s_1 \\
&= \left(\frac{R_{(n)}}{n}\right)m\sum_{l=1}^{m}\left(\frac{|N_l|^2}{\sum_{i=1}^{m}\sum_{j=1}^{|N_i|}\sum_{k=1}^{j} P_k}\right)s_1 \\
&\leq \left(\frac{R_{(n)}}{n}\right)m\sum_{l=1}^{m}\left(\frac{|N_l|^2}{\sum_{j=1}^{|N_l|}\sum_{k=1}^{j} P_k}\right)s_1. \qquad (5.10)
\end{aligned}$$

Now, notice that if $n \to \infty$, for each $l \in \{1,\ldots,m\}$, $|N_l| \to \infty$ almost surely. Consequently, for each l, as $|N_l| \to \infty$, by Lemma 25 (with $W_j \triangleq 1$, $\forall j$) we have that

$$\frac{\sum_{j=1}^{|N_l|}\sum_{k=1}^{j} P_k}{|N_l|^2} \to \frac{\mathbb{E}[P]}{2}$$

and therefore

$$\frac{|N_l|^2}{\sum_{j=1}^{|N_l|}\sum_{k=1}^{j} P_k} \to \frac{2}{\mathbb{E}[P]},$$

almost surely. Consequently, as $n \to \infty$, by applying Lemma 19, we have that Equation (5.10) satisfies

$$\left(\frac{R_{(n)}}{n}\right)m\sum_{l=1}^{m}\left(\frac{|N_l|^2}{\sum_{j=1}^{|N_l|}\sum_{k=1}^{j} P_k}\right)s_1 \to (0)\left(\frac{2m^2 s_1}{\mathbb{E}[P]}\right) = 0,$$

almost surely. Thus, $\lim_{n\to\infty} \frac{Z_n^{SRPR-FM}}{Z_n^*} = 1$ almost surely, which means that SRPR-FM is almost surely asymptotically optimal for the online version of $Q|r_j, pmtn|\sum_j C_j$.

■

5.10.2 Online $P|r_j|\sum_j C_j$

Consider the non-preemptive *Shortest Processing Requirement* (SPR) heuristic, which is also an online algorithm: whenever a machine is available to process a job, choose the job with the shortest processing requirement. Note that the SPR heuristic solves $P||\sum_j C_j$ exactly; see [59]. The reason that we only consider problems with unit weights is that $P||\sum_j w_j C_j$ is NP-hard and our technique for proving asymptotical optimality would break down, since we would require a closed form expression for the optimal cost of this NP-hard problem. Our main result for this section is the following.

Theorem 47 *If* $\mathbb{E}[R] < \infty$, $\mathbb{E}[P] > 0$, *and* $\mathbb{E}[P^2] < \infty$, *then the SPR heuristic is almost surely asymptotically optimal for the online version of* $P|r_j|\sum_j C_j$.

The proof of Theorem 47 is very similar to that for Theorem 46; we highlight only the *differences*.

Proof First, note that any part of the proof relating to the Q class of machines is no longer needed. Next, note that $Z_n^{\mathcal{R}}$ is the optimal solution for $P||\sum_j C_j$. Equation (5.7) is modified to become

$$Z_n^{SPR} \leq n\left(R_{(n)} + P_{(n)}\right) + Z_n^{\mathcal{R}}.$$

The reason for this modification is because at time $R_{(n)}$, we can not relate SPR's actions to the problem $P||\sum_j C_j$, since some machines might be busy processing some jobs. But after $P_{(n)} = \max_{1\leq i\leq n} P_i$ time, we are certain that the machines have finished whatever jobs had been in progress at time $R_{(n)}$. Therefore, at time $R_{(n)} + P_{(n)}$, assuming that no job has been processed, SPR "sees" the problem $P||\sum_j C_j$. The only other significant modification is the final decomposition, which results in:

$$\left(\frac{R_{(n)}}{n} + \frac{P_{(n)}}{n}\right) m \sum_{i=1}^{m} \left(\frac{|N_i|^2}{\sum_{j=1}^{|N_i|}\sum_{k=1}^{j} P_k}\right) \to (0+0)\left(\frac{2m^2}{\mathbb{E}[P]}\right) = 0,$$

almost surely. The rest of the proof remains identical. ■

Chapter 6

Computational Results

6.1 Introduction

The research presented in this thesis thus far has been theoretical in nature. This chapter is intended to provide a complementary computational study of a subset of the algorithms studied previously. There are a number of interesting questions that simulation can answer. First, we can investigate properties of the competitive ratio. For example, is the worst case ratio usually or rarely attained in practice? We can also investigate the average performance of an algorithm. We can exhibit concretely the results of Chapter 5. We are able to see very precisely the speed of convergence to optimality. Under certain stochastic inputs, the convergence is extremely fast. We also consider a scenario where we were unable to prove an asymptotic optimality result: Algorithm SIOR under i.i.d. release dates. Under this scenario, we show that SIOR exhibits computational divergence, i.e. unbounded ratio of online to offline costs.

6.2 The Online TSP on $\mathbb{R}^+$

In this section, we present computational results for the online TSP on $\mathbb{R}_+$. We simulate the algorithms MLIB and MRIN under the following probabilistic assumptions: city locations are i.i.d. random variables and the release dates form a general renewal

process. In particular, we let the city locations L_k be exponential with parameter λ_L (so that city locations are *not* bounded) and the city inter-arrival times X_k be exponential with parameter λ_X. Hence, we consider a time Poisson process where the release dates R_k satisfy $R_k = \sum_{i=1}^{k} X_i$ and $R_k = R_{k-1} + X_k$. The city disclosure times $Q_k = F_k R_k$, where the F_k are i.i.d. random variables that are uniformly distributed on $[0, 1]$.

We now explain our simulation setting: We let ρ_n equal the *actual* ratio of the online cost to the optimal offline cost. We estimate the expected value of ρ_n for a range of cities. In particular, we plot our estimate $\overline{\rho}_n$ of $\mathbb{E}[\rho_n]$ versus the number of cities n for $n = 1, \ldots, 250$. In order to find the estimate $\overline{\rho}_n$, for each n we run the MLIB and MRIN algorithms on 2500 random problem instances, and then take the average.

With respect to the precision of our computational experiments, the standard error for $\overline{\rho}_n$ is defined as $\sigma_{\overline{\rho}_n} = \frac{\sigma_{\rho_n}}{\sqrt{2500}}$. Unfortunately, we do not know σ_{ρ_n}, but we can bound it in the following way. Note that ρ_n is a random variable that is restricted to be within the interval $[1, \frac{3}{2}]$. It is a simple exercise to see that the variance of ρ_n is maximized when ρ_n is equal to either 1 or $\frac{3}{2}$, each with probability $\frac{1}{2}$. The corresponding maximum standard deviation is $\frac{1}{4}$. Thus, $\sigma_{\overline{\rho}_n} \leq \frac{1}{4\sqrt{2500}} = 0.005$. Note that this bound on the precision is a conservative one.

We present an array of results which consists of three graphs that each plot our estimates of $\overline{\rho}_n$ for MLIB and MRIN together, for a variety of values of λ_L and λ_X. Let us consider Figure 6-1, where in all cases, $\mathbb{E}[X] = 1$. We first notice, that in all cases, the online algorithms are asymptotically optimal. We then notice, that for all three choices of parameters, MLIB's online cost converges to the optimal offline cost faster than that of MRIN, supporting our believe that MLIB typically outperforms MRIN. For the case where $\mathbb{E}[L] = 10$, we see that both online costs seem to converge for $1 \leq n < 20$. For $20 \leq n \leq 100$, MRIN's performance deteriorates. MLIB's performance is also slightly perturbed for this range, but not so much as MRIN. For $n > 100$, MRIN's performance resumes converging. Note that MLIB's performance is much more robust and does not deteriorate very much at all. For the case where

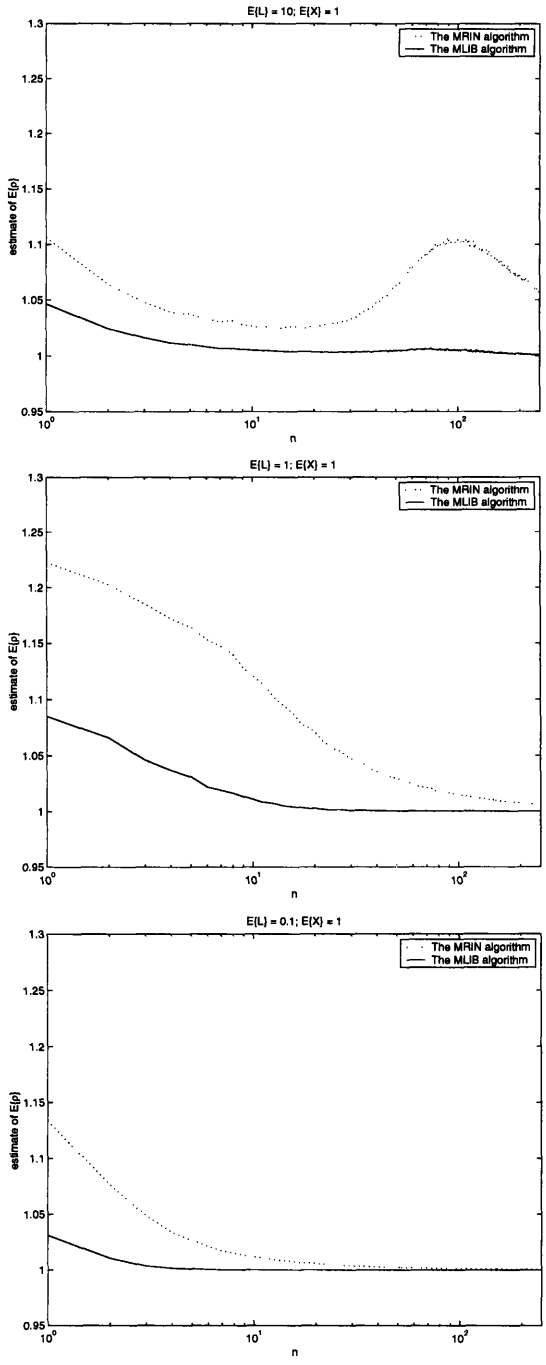

Figure 6-1: Plots of $\overline{\rho}_n$ versus n. In all graphs, $\mathbb{E}[X] = 1$. On the top, $\mathbb{E}[L] = 10$; in the middle, $\mathbb{E}[L] = 1$; on the bottom, $\mathbb{E}[L] = 0.1$.

$\mathbb{E}[L] = 0.1$, the convergence for both algorithms is very fast; for $n = 10$, both online costs are very nearly equal to the offline cost. Finally, we consider the case $\mathbb{E}[L] = 1$. Note that for n small, the performance of both MRIN and MLIB is worse than their respective performances in the other two cases. Otherwise, the convergence is rather fast for this final case.

Let us conclude with a discussion of this intriguing phenomenon, where the initial ($n = 1$) ratios are largest when $\mathbb{E}[L] = 1$ (i.e., $\lambda_L = 1$) and decrease when we either increase *or* decrease λ_L. This phenomenon can be explained by noting that the inter-arrival times have the same parameter value: $\lambda_X = 1$. Consider the following instance motivated by the equality of the parameters λ_L and λ_X: $(l, r, q) = (1, 1, 1)$. The online cost of MLIB and MRIN on this instance is 3 while the optimal offline cost is 2. This is a worst case instance. Consequently, the equality of the above parameters is reasonably seen to be the culprit for the high initial ratios of online to offline cost. Using a similar realization argument, one can argue that the ratio will be lower in the other cases. In conclusion, these results have showed that, for *all* combinations of parameters considered, MLIB outperformed MRIN on average.

6.2.1 On Rates of Convergence and Regime Behaviors

While we have shown computationally that the online costs converge to the optimal offline costs, we have not provided any rates nor descriptions of the convergence. To proceed further, we consider three regimes: (1) $\mathbb{E}[L] \gg \mathbb{E}[X]$, (2) $\mathbb{E}[L] \ll \mathbb{E}[X]$ and (3) $\mathbb{E}[L] \approx \mathbb{E}[X]$. For each of these regimes, we consider behaviors for n "small," "moderate" and "large."

In Figures 6-2, 6-3 and 6-4, we present sample trajectories for both algorithms under each of these nine cases. Distributional parameters are evident from the graphs in Figures 6-2–6-4. Note that for n to be classified as either small, moderate or large is subjective. We have attempted to choose values so that the resulting sample trajectories are informative.

We first consider Regime (1). When n is small, we have that $\mathbb{E}[L] \gg \mathbb{E}[R_n] = n\mathbb{E}[X]$. In this case, as the salesmen are traveling to the first disclosed/released city,

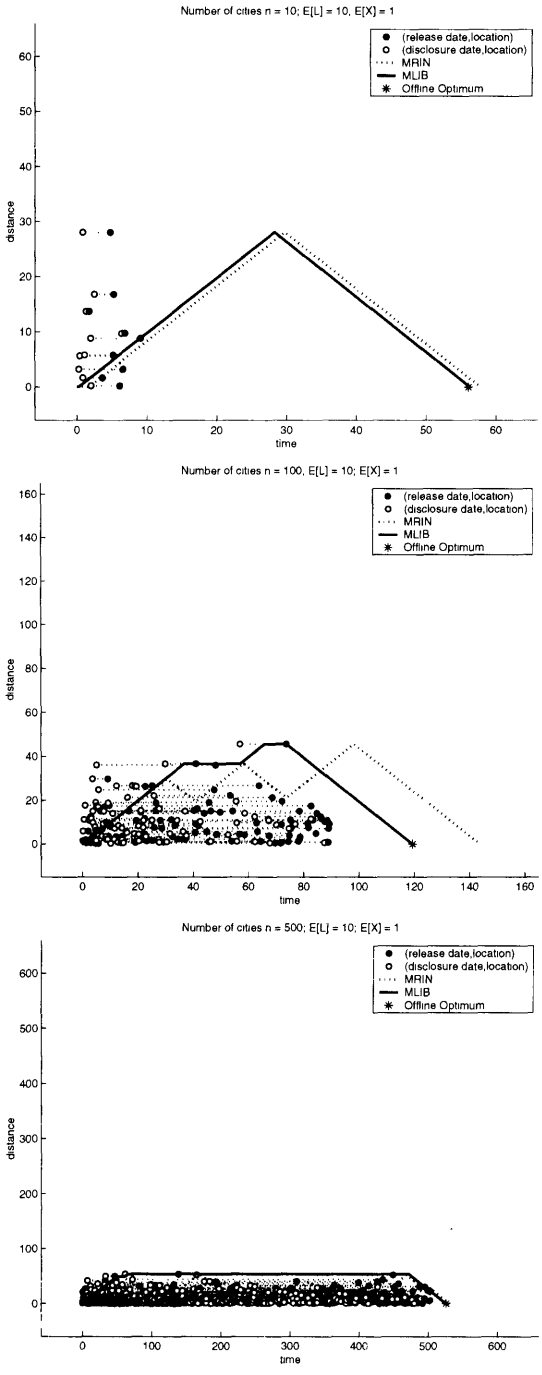

Figure 6-2: Behavioral Regime 1

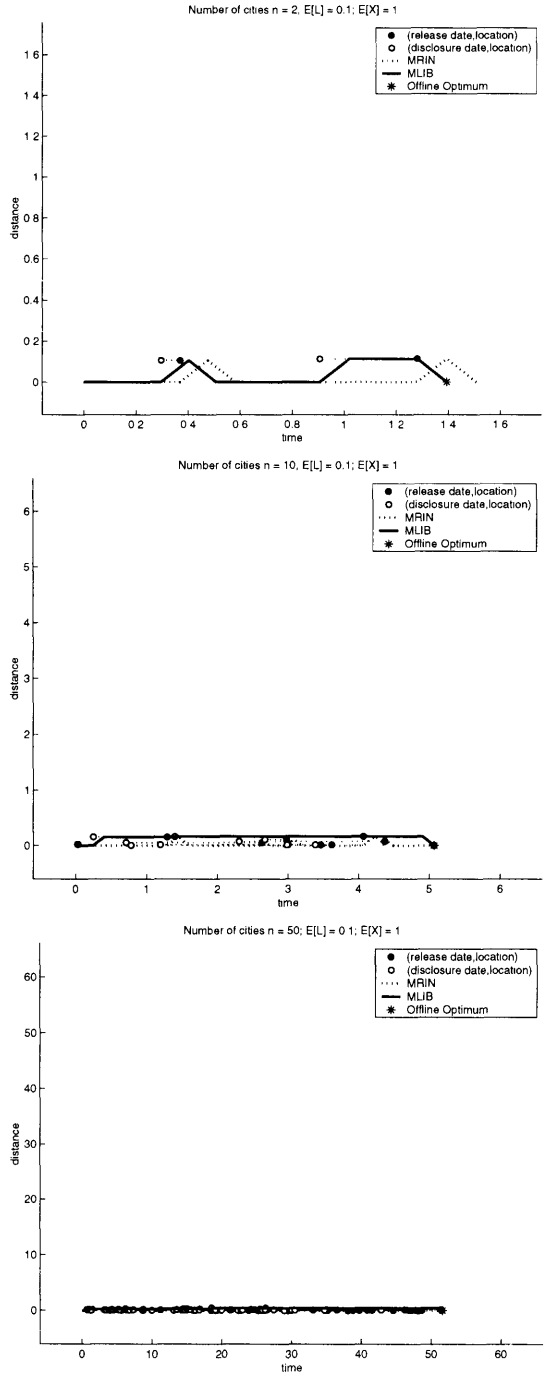

Figure 6-3: Behavioral Regime 2

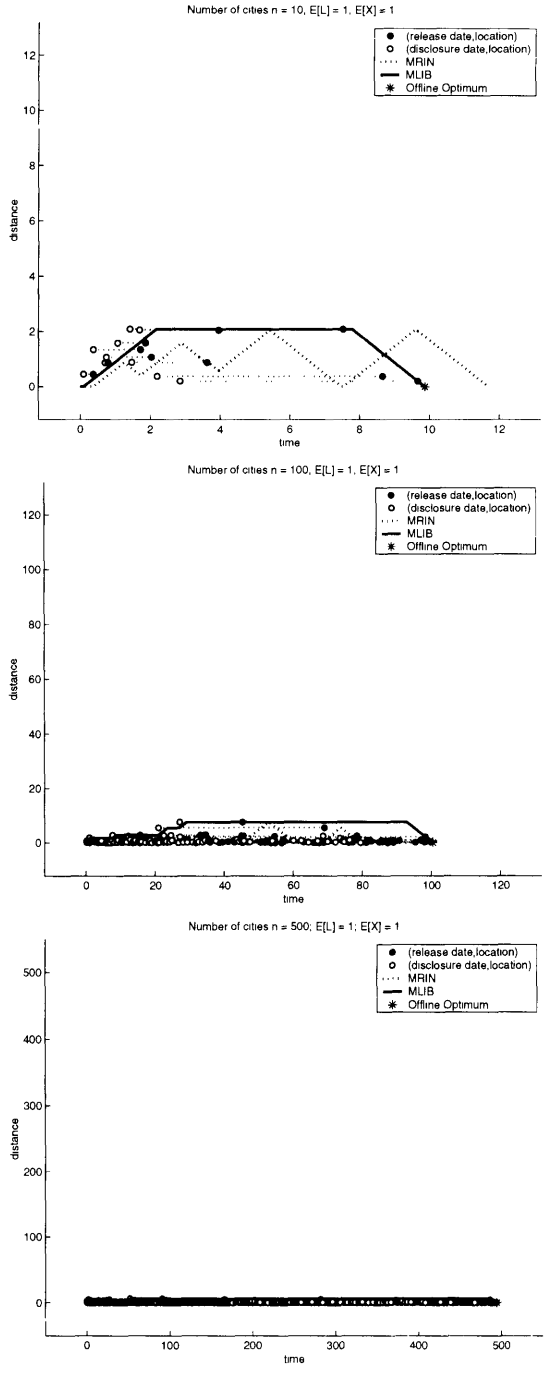

Figure 6-4: Behavioral regime 3.

all other cities are released. The resulting trajectory is to simply go to the maximum city location L_{max} and back. Due to the disclosure dates, MLIB is expected to have a slight advantage. The costs of both online algorithms as well as the optimal offline algorithm are approximately $2L_{max}$. Thus, in this case, we expect that MLIB and MRIN perform rather well with respect to the optimal offline algorithm. When n is moderate, L_{max} is on the same order as R_n. In this case, it is not clear what will happen. However, a single city instance where $R_1 = L_1$ induces the worst possible online cost for MRIN: $Z_1^{\text{MRIN}} = \frac{3}{2}Z_1^*$. Consequently, we are led to believe that in this sub-regime the online algorithms (at least MRIN) will not perform very well. We saw computationally that MRIN does not perform well for this sub-regime, but MLIB is more robust and performs rather well. When n is large, a limit argument applies: L_{max} is small with respect to R_n and the online costs converge to the optimal offline cost. To summarize, under Regime (1) conditions, we expect the online salesmen to perform well for n small and n large, while for n moderate, the online performances of MRIN and MLIB might deteriorate; this agrees with our computational studies. Figure 6-2 corresponds to Regime (1).

Considering Regime (2), we expect that both MRIN and MLIB salesmen would return to the origin between visiting each city. This situation is equivalent to a series of one-city instances. Note that in this case, MLIB is expected to always outperform MRIN. However, we also see that both online costs and the optimal offline cost are approximately equal to R_n. Even though MLIB is expected to always outperform MRIN in this case, both algorithms will perform well with respect to the optimal offline algorithm. Thus, we expect rather good performance overall under Regime (2). Figure 6-3 corresponds to Regime (2).

Finally, the behavior of Regime (3) can be seen to be similar to Regime (1) under moderate n, but for Regime (3) the behavior occurring for n small. For larger n, the same limit argument applies. Figure 6-4 corresponds to Regime (3).

6.3 The Online TSP on $[0,1]^2$

The majority of our simulations for the online TSP consider algorithm PAH (which has a competitive ratio of 2). This algorithm is appealing because its main subroutine calls for solving a classic TSP. For this subroutine, we utilize the powerful Concorde [5] TSP solver. Consequently, these results are of a practical interest.

6.3.1 Fast Asymptotic Optimality

We consider the following probabilistic situations. City locations are uniformly distributed on the unit square $[0,1]^2$. We consider a specific generator for each of the release date structures. We first simulate the case where city release dates are uniformly distributed on $[0,1]$ and then we simulate the case where the release dates are generated from a Poisson process of parameter 1. For each value of n, we simulate 20 trials and then plot the average ratio of the cost of algorithm PAH to a lower bound on the optimal offline cost: $\rho_n \triangleq \frac{Z_n^{PAH}}{\max\{R_n, L_{TSP}\}}$; therefore, the plots are *conservative.* We also superimpose polynomial functions on the simulation results, which agree with the predicted rate of convergence. These ratios are presented in the top and bottom plots of Figure 6-5, respectively.

We now briefly discuss the precision of our simulation results. Clearly $\rho_n \geq 1$. It can be be shown that $\rho_n \leq 3$. Noting that ρ_n is a random variable, it can be seen that the standard deviation of ρ_n is maximized, equaling 1, when $\rho_n \in \{1,3\}$, each with probability $\frac{1}{2}$. Our simulation studies estimate the expected value of ρ_n, so the standard error of our estimate, using 20 trials, can be bounded: $\sigma_{\overline{\rho_n}} \triangleq \frac{\sigma_{\rho_n}}{\sqrt{20}} \leq \frac{1}{\sqrt{20}} < 0.23$. Finally, note that these bounds are *conservative.*

6.3.2 Intriguing Behaviors

We now consider the situation where the city locations are uniformly distributed on $[0,1]^2$ and the release dates are uniformly distributed on $[0,c]$, for $c \in \{5,8,10\}$. In Figure 6-6, we see some intriguing sample paths.

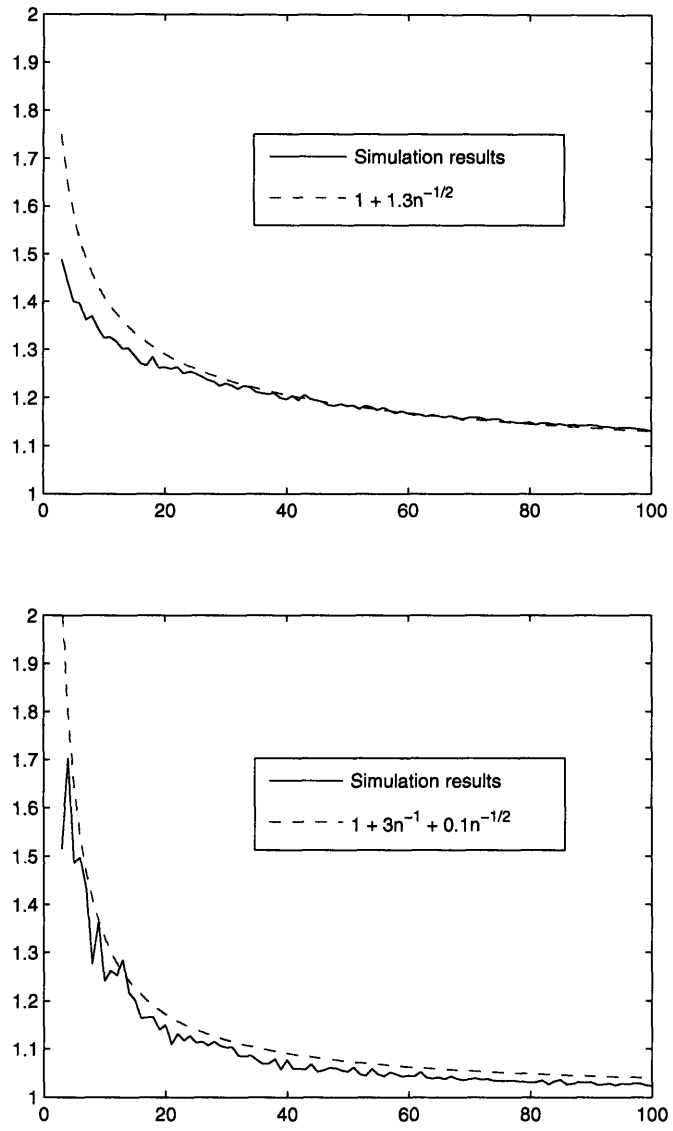

Figure 6-5: Upper bounds on the ratios of the cost of PAH to the optimal offline cost, as a function of n. Each data point is the average of 20 trials. The top plot considers release dates that are uniformly distributed on $[0, 1]$. The bottom plot considers release dates that are generated from a Poisson process of unit parameter.

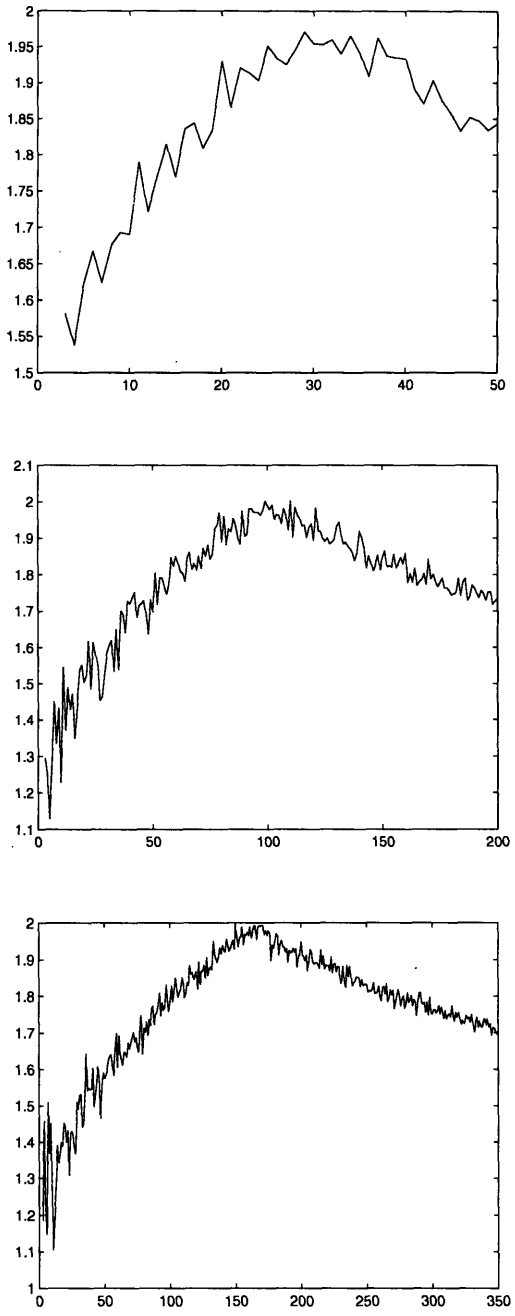

Figure 6-6: Upper bounds on the ratios of the cost of PAH to the optimal offline cost, as a function of n. The top, middle and bottom plots consider release dates that are uniformly distributed on $[0, 5]$, $[0, 8]$ and $[0, 10]$, respectively. These plots consist of single sample paths.

We now offer an explanation of this behavior; we focus our discussion on the case where the release dates are uniformly distributed on $[0, 10]$; a similar argument applies to the other cases. Note that the peak occurs when the number of cities is approximately $n \approx 175$. We assume that for this value of n, the TSP tour behaves asymptotically as predicted by the Beardwood, Halton and Hammersley [12] result. Thus,

$$L_{TSP}(n) \approx \beta\sqrt{n} \approx 10.$$

Notice that for $n = 175$, $R_n \approx 10$ also. So at the peak, the final release date is approximately equal to the optimal (classic) TSP tour length; i.e., $R_n \approx L_{TSP}$; we also believe that this is the reason for the peak. Now, note that the plots in Figure 6-6 were produced by outputting the ratio of PAH to the lower bound on Z_n^*:

$$\frac{Z_n^{PAH}}{\max\{R_n, L_{TSP}(n)\}}.$$

Therefore, computationally, the cost of PAH at the peak is approximately equal to $R_n + L_{TSP}$, which is similar to upper bounds that have been proved; this (more or less) explains the ratio of approximately 2. The reason we get a *peak* is because of the asymptotic behavior of R_n and L_{TSP} as a function of n. We know that $L_{TSP} = \Theta(\sqrt{n})$ a.s. For i.i.d. release dates uniform on $[0, c]$, for some $c > 0$, we have that $R_n = \Theta(1)$ a.s.; for renewal process release dates, we have that $R_n = \Theta(n)$ a.s. This explains the peak and also explains why we sometimes see a peak and at other times we don't.

Finally, we mention that we see similar behaviors in other scenarios. When the release dates are generated from a Poisson process of appropriate parameter, we see similar behaviors. Additionally, these behaviors are not dependent on the algorithm PAH – we see similar behaviors for algorithm GMP as well.

6.4 The Online TRP on $[0, 1]^2$

We now turn to the online TRP and we simulate algorithm SIOR.

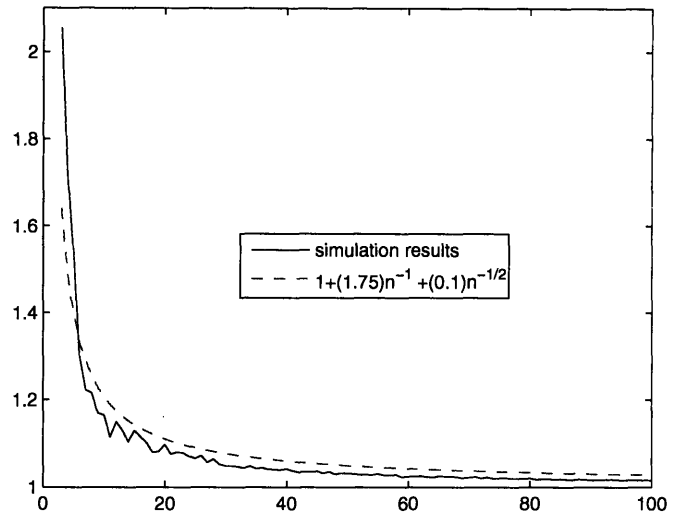

Figure 6-7: Upper bounds on the ratios of the cost of SIOR to the optimal offline cost, as a function of n. Each data point is the average of 20 trials.

6.4.1 Fast Asymptotic Optimality

We again consider the probabilistic scenario where city locations are uniformly distributed on the unit square $[0,1]^2$ and the city release dates are generated from a Poisson process of unit parameter. We consider the case where $w_i = 1$, $\forall i$. For each value of n, we simulate 20 trials and then plot the average ratio of the cost of algorithm SIOR to a lower bound on the optimal offline cost: $\frac{Z_n^{SIOR}}{\sum_{i=1}^{n} R_i}$; therefore, the plots are *conservative*. We also superimpose polynomial functions on the simulation results. We see that the rate of convergence is as predicted by the theory ($\frac{1}{\sqrt{n}}$); furthermore, it is interesting to note that the actual speed of convergence for SIOR (an online TRP algorithm) is faster than that for PAH (an online TSP algorithm) under the same probabilistic setup.

Unfortunately, we are unable to say much regarding the precision of these experiments since we do not have a constant upper bound on $\frac{Z_n^{SIOR}}{\sum_{i=1}^{n} R_n}$. However, we do believe our results are very accurate. These ratios are presented in Figure 6-7.

6.4.2 Divergence

To conclude our computational studies on online routing problems, we consider algorithm SIOR under the following probabilistic assumptions: City locations are uniformly distributed on the unit square $[0,1]^2$ and the city release dates are uniformly distributed on $[0,1]$. Recall that we were unable to attain an asymptotic optimality result for algorithm SIOR under these assumptions. Computationally, in Figure 6-8, we see a divergence of the upper bounds on the ratios of online to offline costs (nothing can be said of the actual ratios). Consequently, our proof technique will not suffice in this situation. We suspect it is not possible to attain an asymptotic optimality result under these conditions.

We now offer an explanation of this phenomenon. Considering algorithm SIOR, we are able to write the completion times of the cities in recursive form. Clearly, $C_1 = R_1 + d(o, L_1)$. By the definition of SIOR, for $k > 1$, $C_k = \max\{C_{k-1}, R_k\} + d(L_{k-1}, L_k)$. Since, in the current situation, the release dates and city locations are comparably distributed, we expect that $C_k > R_k$ for all k. Therefore, if this effect compounds, we see that Z_n^{SIOR} will grow faster than $\sum_{i=1}^{n} R_i$ and this would explain the divergence. Returning to the case where the release dates are generated from a Poisson process, the convergence can be intuitively explained by noticing that for large enough k, $C_k \approx R_k$, since the release date terms will dominate the costs.

6.5 Online Machine Scheduling Problems

To conclude this section, we present a computational study of the online machine scheduling algorithms from Chapter 3 when the problem data is randomly generated. To the best of our knowledge, this study is one of the first computational investigations of LP-based scheduling algorithms; particularly, we believe it is the first that considers *online* algorithms as well as the first to consider *multiple* machines. We are aware of only one other paper [62] which investigates a similar topic. These authors perform an extensive computational study of a number of heuristics and approximation algorithms for the single machine problem $1|r_j|\sum_j w_j C_j$ (the offline version).

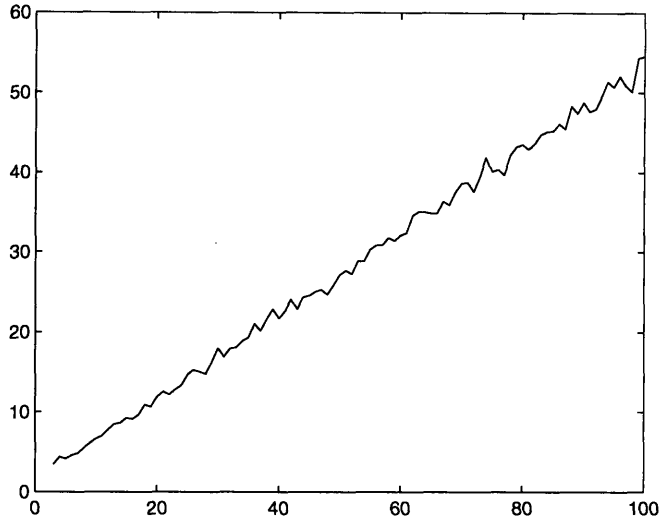

Figure 6-8: Upper bounds on the ratios of the cost of SIOR to the optimal offline cost, as a function of n. This output is a single sample path.

As in their work, our results suggest that the practical performance of LP-based scheduling algorithms is much better than what theory predicts.

We let the data for each job be *independent* realizations of uniformly distributed random variables: $r_i \sim U[0, R]$, $p_i \sim U[0, P]$ and $w_i \sim U[0, W]$, for $i = 1, \ldots, n$.

We now describe our general approach. We first fix the parameters R, P and W and vary m and n. Then, we fix m and n and vary R, P and W. For each set of parameters we run 1000 trials to give the mean, max and standard deviation (presented in this order) of the ratio of the cost of the online algorithm (NAS, NASR or PASR) to the lower bound given in Lemma 1.

6.5.1 NAS

We first fix $R = P = W = 10$ and study the effect of changing m and n.

	$n = 10$	$n = 100$	$n = 500$
$m = 1$	(1.2226, 1.4321, 0.0421)	(1.0283, 1.0433, 0.0019)	(1.0056, 1.0063, 0.0001)
$m = 10$	(1.3275, 1.5293, 0.0559)	(1.1579, 1.1842, 0.0063)	(1.0421, 1.0449, 0.0009)
$m = 25$	(1.3308, 1.5548, 0.0614)	(1.2613, 1.2866, 0.0076)	(1.0871, 1.0943, 0.0017)

Next, we fix $P = W = 10$ and let R depend on n; specifically, we let $R = n$ for $n \in \{10, 100, 500\}$. We also vary m as before: $m \in \{1, 10, 25\}$.

	$R = n = 10$	$R = n = 100$	$R = n = 500$
$m = 1$	(1.2226, 1.4321, 0.0421)	(1.0463, 1.0958, 0.0111)	(1.0131, 1.0252, 0.0031)
$m = 10$	(1.3275, 1.5293, 0.0559)	(1.0526, 1.0677, 0.0045)	(1.0112, 1.0128, 0.0005)
$m = 25$	(1.3308, 1.5548, 0.0614)	(1.0521, 1.0674, 0.0045)	(1.0110, 1.0123, 0.0004)

Finally, we fix $m = 10$ and $n = 100$ (the middle case) and vary R, P and W in the following set $\{1, 10\}$; we present the 8 results below.

	$(R, P) = (1, 1)$	$(R, P) = (1, 10)$
$W = 1$	(1.1057, 1.1139, 0.0026)	(1.1579, 1.1776, 0.0053)
$W = 10$	(1.1057, 1.1157, 0.0025)	(1.1583, 1.1784, 0.0054)

	$(R, P) = (10, 1)$	$(R, P) = (10, 10)$
$W = 1$	(1.0799, 1.0915, 0.0040)	(1.1578, 1.1876, 0.0061)
$W = 10$	(1.0799, 1.0928, 0.0041)	(1.1581, 1.1816, 0.0061)

6.5.2 NASR

We first fix $R = P = W = 10$ and study the effect of changing m and n. The expected competitive ratio for $m = 1$, $m = 10$ and $m = 25$ are 1.6853, 1.9673 and

1.9869, respectively.

	$n = 10$	$n = 100$	$n = 500$
$m = 1$	(1.2140, 1.5575, 0.0569)	(1.0336, 1.0429, 0.0026)	(1.0075, 1.0083, 0.0002)
$m = 10$	(1.3186, 1.5094, 0.0574)	(1.1536, 1.1849, 0.0065)	(1.0415, 1.0449, 0.0009)
$m = 25$	(1.3331, 1.6013, 0.0607)	(1.2574, 1.2835, 0.0081)	(1.0863, 1.0918, 0.0016)

Next, we fix $P = W = 10$ and let R depend on n; specifically, we let $R = n$ for $n \in \{10, 100, 500\}$. We also vary m as before: $m \in \{1, 10, 25\}$.

	$R = n = 10$	$R = n = 100$	$R = n = 500$
$m = 1$	(1.2140, 1.5575, 0.0569)	(1.0449, 1.0903, 0.0098)	(1.0121, 1.0238, 0.0030)
$m = 10$	(1.3186, 1.5094, 0.0574)	(1.0517, 1.0717, 0.0044)	(1.0109, 1.0124, 0.0004)
$m = 25$	(1.3331, 1.6013, 0.0607)	(1.0516, 1.0685, 0.0045)	(1.0109, 1.0123, 0.0005)

Finally, we fix $m = 10$ and $n = 100$ and vary R, P and W in the following set $\{1, 10\}$; we present the 8 results below.

	$(R, P) = (1, 1)$	$(R, P) = (1, 10)$
$W = 1$	(1.1080, 1.1190, 0.0028)	(1.1553, 1.1711, 0.0053)
$W = 10$	(1.1079, 1.1207, 0.0027)	(1.1551, 1.1739, 0.0052)

	$(R, P) = (10, 1)$	$(R, P) = (10, 10)$
$W = 1$	(1.0808, 1.0945, 0.0046)	(1.1535, 1.1766, 0.0065)
$W = 10$	(1.0812, 1.0951, 0.0047)	(1.1533, 1.1772, 0.0066)

6.5.3 PASR

We first fix $R = P = W = 10$ and study the effect of changing m and n. The expected competitive ratio for $m = 1$, $m = 10$ and $m = 25$ are 1.3333, 1.8961 and 1.9595, respectively.

	$n = 10$	$n = 100$	$n = 500$
$m = 1$	(1.0887, 1.5016, 0.0650)	(1.0015, 1.0091, 0.0013)	(1.0000, 1.0004, 0.0000)
$m = 10$	(1.2678, 1.3957, 0.0461)	(1.0430, 1.0756, 0.0110)	(1.0015, 1.0038, 0.0006)
$m = 25$	(1.3081, 1.4914, 0.0559)	(1.1515, 1.2112, 0.0160)	(1.0117, 1.7732, 0.0242)

Next, we fix $P = W = 10$ and let R depend on n; specifically, we let $R = n$ for $n \in \{10, 100, 500\}$. We also vary m as before: $m \in \{1, 10, 25\}$.

	$R = n = 10$	$R = n = 100$	$R = n = 500$
$m = 1$	(1.0887, 1.5016, 0.0650)	(1.0107, 1.0595, 0.0071)	(1.0028, 1.0189, 0.0019)
$m = 10$	(1.2678, 1.3957, 0.0461)	(1.0426, 1.0540, 0.0037)	(1.0090, 1.0105, 0.0004)
$m = 25$	(1.3081, 1.4914, 0.0559)	(1.0486, 1.0634, 0.0040)	(1.0102, 1.0114, 0.0004)

Finally, we fix $m = 10$ and $n = 100$ and vary R, P and W in the following set $\{1, 10\}$; we present the 8 results below.

	$(R, P) = (1, 1)$	$(R, P) = (1, 10)$
$W = 1$	(1.0037, 1.3452, 0.0108)	(1.0065, 1.0095, 0.0007)
$W = 10$	(1.0034, 1.0048, 0.0003)	(1.0065, 1.0094, 0.0007)

	$(R, P) = (10, 1)$	$(R, P) = (10, 10)$
$W = 1$	(1.0220, 1.0662, 0.0029)	(1.0433, 1.5628, 0.0199)
$W = 10$	(1.0219, 1.0316, 0.0023)	(1.0436, 1.0815, 0.0117)

6.5.4 Observations

- For all three algorithms, as m increases, the mean and max increase.
- For all three algorithms, as n increases, the mean and max decrease.
- Statistically, it seems that NAS and NASR are comparable.
- Over 1000 trials, the max ratio of NASR never exceeded the expected competitive ratio.
- Over 1000 trials, the max ratio of PASR did exceed the expected competitive ratio (e.g., $m = 1$, $n = 10$).

Chapter 7

Conclusion and Future Work

In this chapter we first summarize the thesis and then we briefly discuss some open problems and directions for future research.

7.1 Summary of Thesis

In Chapter 1 we began by motivating online optimization in routing and scheduling. We then gave an overview of online optimization and methods for analyzing online algorithms. Next, we gave a literature review covering relevant work in online routing and scheduling. Finally, we gave an outline of the thesis which detailed the main contributions.

In Chapter 2 we considered versions of the online TSP, TRP and k-TSP. We began by giving a new polynomial-time online algorithm for the online TSP. We then introduced the online TSP with precedence and capacity constraints and designed an algorithm for this problem. Next, we introduced the online TSP with multiple vehicles and designed an algorithm for this problem as well. Finally, we gave the definitions of the online TRP and k-TSP.

In Chapter 3 we considered online machine scheduling problems. We began by designing and analyzing a deterministic online algorithm for the $P|r_j|\sum w_jC_j$ problem. Next, we designed and analyzed randomized online algorithms for the $P|r_j|\sum w_jC_j$ and $P|r_j, pmtn|\sum w_jC_j$ problems.

In Chapter 4 we studied resource augmentation for online routing problems. We first studied information augmentation, in the form of disclosure dates; we applied disclosure dates to the online TSP, online TSP with multiple vehicles and the online TRP. We then considered speed augmentation for the online TSP with precedence and capacity constraints. Next, we studied speed and vehicle augmentation for the online TSP with multiple servers. Finally, we studied speed and constraint augmentation for the online k-TSP.

In Chapter 5 we applied stochastic asymptotic analysis to a variety of online routing and scheduling problems. We first proved an almost sure capacity augmentation result for the online TSP with capacity constraints. We then gave algorithms that are almost surely asymptotically optimal for the online TSP with precedence constraints, the online TSP with capacity constraints and the online TRP with precedence constraints. We also characterized the rate of convergence to optimality for these algorithms. Finally, we proved similar results for a number of online machine scheduling algorithms.

In Chapter 6 we provided Monte Carlo simulation results for the online TSP, online TRP and numerous online machine scheduling problems.

7.2 Future Research

There are a number of directions that the research in this thesis can be extended. We mention a few here.

- Consider the online TSP. The first interesting question to ask is what happens if we are not required to visit all cities in an instance. In other words, we allow for accept and reject decisions. This creates new difficulties however – it is easy to create (small) instances that induce an infinite competitive ratio. Therefore, we are at a loss for how to evaluate algorithms that allow for accept and reject decisions. New measures are needed and this is an interesting research question in its own right. If successful in creating a meaningful measure, it would also make sense to introduce prices for the cities. In other words, we can ask at

what price are we willing to visit a city. This opens the door to online revenue management. Also, clearly, this approach is not only limited to the online TSP.

- An open problem of interest to the author is the competitive ratio for the online TRP. In [28] and [51], respectively, a lower bound of $(1+\sqrt{2}) \approx 2.41$ and an upper bound of of $(1+\sqrt{2})^2 \approx 5.83$ are given for the competitive ratio of the online TRP. The author believes that the greedy strategy GL (simplified appropriately for the online TRP) defined in Chapter 5 has a competitive ratio of 3, but a rigorous proof has been elusive. The author also believes that simple modifications to GL can result in a competitive ratio of $(1+\sqrt{2})$, a best-possible online algorithm.

- Another interesting idea is to combine an online problem explicitly [1] with game theory. Consider the situation where we have multiple salesmen that *compete* for cities. Whichever salesman arrives first to a dynamically revealed city receives a reward for that city. Interesting questions can be asked: What is an optimal strategy? Does a Nash Equilibrium exist? This research problem was first discussed with Nicolas Stier.

- Along the lines of the disclosure dates studied in Chapter 4, we can study questions about the value of varying degrees of information about the problem instance. Consider the online TSP where each city i requires service that takes a certain amount of time s_i. Many different online problems can be defined: (1) s_i is revealed at a city's release date, (2) Arrival at a city reveals s_i, (3) Arrival at a city reveals a probabilistic distribution for s_i and (4) The value of s_i is not known until the salesman has finished service. Additionally, preemption can also be introduced in a number of ways (e.g., preempt and resume where the salesman left off or preempt and start over). Another variant of the online TSP is to give a distribution for the city location at the city's disclosure date (the city's actual location is revealed at the release date).

[1] The adversarial model in online optimization is implicitly game theoretic.

- Finally, online optimization has traditionally been a tool of computer science and its use in operations research is rather new. It would be interesting to apply online optimization to other classic operations research problems.

Bibliography

[1] F. Afrati, E. Bampis, C. Chekuri, D. Karger, C. Kenyon, S. Khanna, I. Milis, M. Queyranne, M. Skutella, C. Stein, and M. Sviridenko. Approximation schemes for minimizing average weighted completion time with release dates. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 32–43, 1999.

[2] L. Allulli, G. Ausiello, and L. Laura. On the power of lookahead in on-line vehicle routing problems. In *Proceedings of the Eleventh International Computing and Combinatorics Conference*, pages 728–736, 2005.

[3] K. Altinkemer and B. Gavish. Heuristics for delivery problems with constant error estimates. *Transportation Science*, 24:295–297, 1990.

[4] E. Anderson and C. Potts. On-line scheduling of a single machine to minimize total weighted completion time. *Mathematics of Operations Research*, 29(3):686–697, 2004.

[5] D. Applegate, R. Bixby, V. Chvtal, and W. Cook. Concorde software for solving tsp. http://www.tsp.gatech.edu/concorde.html.

[6] S. Arora. Polynomial-time approximation schemes for euclidean tsp and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998.

[7] N. Ascheuer, M. Grotschel, and J. Rambau. Combinatorial online optimization in practice. *Optima*, 57, 1998.

[8] G. Ausiello, L. Allulli, V. Bonifaci, and L. Laura. On-line algorithms, real time, the virtue of laziness, and the power of clairvoyance. working paper, University of Rome "La Sapienza", January 2006.

[9] G. Ausiello, M. Demange, L. Laura, and V. Paschos. Algorithms for the on-line quota traveling salesman problem. *Information Processing Letters*, 92(2):89–94, 2004.

[10] G. Ausiello, E. Feuerstein, S. Leonardi, L. Stougie, and M. Talamo. Algorithms for the on-line travelling salesman. *Algorithmica*, 29(4):560–581, 2001.

[11] V. Barzov. Urop research summary. working paper, MIT Operations Research Center, 2004.

[12] J. Beardwood, J. Halton, and J. Hammersley. The shortest path through many points. *Proceedings of the Cambridge Philosophical Society*, 55:299–327, 1959.

[13] S. Ben-David and A. Borodin. A new measure for the study of on-line algorithms. *Algorithmica*, 11(1):73–91, 1994.

[14] D. Bertsimas. *Probabilistic Combinatorial Optimization Problems.* PhD thesis, Massachusetts Institute of Technology, 1988.

[15] M. Blom, S. Krumke, W. de Paepe, and L. Stougie. The online tsp against fair adversaries. *INFORMS Journal on Computing*, 13(2):138–148, 2001.

[16] A. Blum, S. Chawla, D. Karger, T. Lane, A. Meyerson, and M. Minkoff. Approximation algorithms for orienteering and discounted-reward tsp. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 46–55, 2003.

[17] A. Bompadre, M. Dror, and J. Orlin. Probabilistic analysis of unit demand vehicle routing problems. working paper, MIT Operations Research Center, September 2004.

[18] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, first edition, 1998.

[19] J. Bramel, E. Coffman, P. Shor, and D. Simchi-Levi. Probabilistic analysis of the capacitated vehicle routing problem with unsplit demands. *Operations Research*, 40(6):1095–1106, 1992.

[20] S. Chakrabarti, C. Phillips, A. Schulz, D. Shmoys, C. Stein, and J. Wein. Improved scheduling algorithms for minsum criteria. In *Automata, Languages and Programming (ICALP)*, pages 646–657. Springer LNCS 1099, 1996.

[21] K. Chaudhuri, B. Godfrey, S. Rao, and K. Talwar. Paths, trees and minimizing latency. In *Proceeding of the 44th Annual IEEE Symposium on Foundations of Computer Science*, 2003.

[22] C. Chekuri, R. Motwani, B. Natarajan, and C. Stein. Approximation techniques for average completion time scheduling. *SIAM Journal on Computing*, 31:146–166, 2001.

[23] C. Chou, H. Liu, M. Queyranne, and D. Simchi-Levi. On the asymptotic optimality of a simple on-line algorithm for the stochastic single machine weighted completion time problem and its extensions. *Operations Research*, 2005. To appear.

[24] C. Chou, M. Queyranne, and D. Simchi-Levi. The asymptotic performance ratio of an on-line algorithm for uniform parallel machine scheduling with release dates. *Mathematical Programming*, 2005. To appear.

[25] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Management Sciences Research Report 388, Carnegie-Mellon University, February 1976.

[26] J. Correa and M. Wagner. Lp-based online scheduling: from single to parallel machines. In *Proceedings of the 11th Integer Programming and Combinatorial Optimization Conference (IPCO)*, pages 196–209. Springer LNCS 3509, 2005.

[27] W. Eastman, S. Even, and I. Isaacs. Bounds for the optimal scheduling of n jobs on m processors. *Management Science*, 11:268–279, 1964.

[28] E. Feuerstein and L. Stougie. On-line single-server dial-a-ride problems. *Theoretical Computer Science*, 268(1):91–105, 2001.

[29] A. Fiat and G. Woeginger. *Online Algorithms: The State of the Art.* Springer Verlag LNCS State of the Art Survey, 1998.

[30] G. Frederickson, M. Hecht, and C. Kim. Approximation algorithms for some routing problems. *SIAM Journal on Computing*, 7(2):178–193, 1978.

[31] J. Galambos. *The Asymptotic Theory of Extreme Order Statistics.* Robert E. Krieger Publishing Company, 1987.

[32] B. Gnedenko. Sur la distribution limite du terme maximum d'une série aléatoire. *Annals of Mathematics*, 44:423–453, 1943.

[33] M. Goemans. A supermodular relaxation for scheduling with release dates. In *Proceedings of the 5th Integer Programming and Combinatorial Optimization Conference (IPCO)*, pages 288–300. Springer LNCS 1084, 1996.

[34] M. Goemans. Improved approximation algorithms for scheduling with release dates. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 591–598, 1997.

[35] M. Goemans, M. Queyranne, A. Schulz, M. Skutella, and Y. Wang. Single machine scheduling with release dates. *SIAM Journal on Discrete Mathematics*, 15:165–192, 2002.

[36] R. Graham, E. Lawler, J. Lenstra, and A.H.G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

[37] G. Grimmett and D. Stirzaker. *Probability and Random Processes.* Clarendon Press, second edition, 1992.

[38] M. Haimovich and A.H.G. Rinnooy Kan. Bounds and heuristics for capacitated routing problems. *Mathematics of Operations Research*, 10:527–542, 1985.

[39] L. Hall, A. Schulz, D. Shmoys, and J. Wein. Scheduling to minimize average completion time: off-line and on-line approximation algorithms. *Mathematics of Operations Research*, 22(3):513–544, 1997.

[40] J. Hoogeveen and A.P.A. Vestjens. Optimal on-line algorithms for single-machine scheduling. In *Proceedings of the 5th Integer Programming and Combinatorial Optimization Conference (IPCO)*, pages 404–414. Springer LNCS 1084, 1996.

[41] P. Jaillet. *Probabilistic Traveling Salesman Problems.* PhD thesis, Massachusetts Institute of Technology, 1985.

[42] P. Jaillet and M. Stafford. Online searching. *Operations Research*, 49:501–516, 2001.

[43] P. Jaillet and M. Wagner. Online routing problems: value of advanced information as improved competitive ratios. *Transportation Science*, 2006. To appear.

[44] B. Kalyanasundaram and K. Pruhs. Constructing competitive tours from local information. *Theoretical Computer Science*, 130(1):125–138, 1994.

[45] B. Kalyanasundaram and K. Pruhs. The online transportation problem. In *Proceedings of the European Symposium on Algorithms*, pages 484–493, 1995.

[46] B. Kalyanasundaram and K. Pruhs. Speed is as powerful as clairvoyance. *Journal of the ACM*, 47(4):617–643, 2000.

[47] P. Kaminsky and D. Simchi-Levi. Asymptotic analysis of an on-line algorithm for the single machine completion time problem with release dates. *Operations Research Letters*, 29:141–148, 2001.

[48] A. Karlin, M. Manasse, L. Rudolph, and D. Sleator. Competitive snoopy caching. *Algorithmica*, 3:79–119, 1988.

[49] B. Korte and J. Vygen. *Combinatorial Optimization, Theory and Algorithms.* Springer, second edition, 2002.

[50] E. Koutsoupias and C. Papadimitriou. Beyond competitive analysis. *SIAM Journal on Computing*, 30(1):300–317, 2000.

[51] S. Krumke, W. de Paepe, D. Poensgen, and L. Stougie. News from the online traveling repairman. *Theoretical Computer Science*, 295:279–294, 2003.

[52] S. Krumke, W. de Paepe, D. Poensgen, and L. Stougie. Erratum to "news from the online traveling repairman". *Theoretical Computer Science*, 352:347–348, 2006.

[53] E. Lawler, J. Lenstra, A.H.G. Rinnooy Kan, and D. Shmoys. *The Traveling Salesman Problem, A Guided Tour of Combinatorial Optimization.* John Wiley & Sons Ltd., 1985.

[54] M. Lipmann. The online traveling salesman problem on the line. Master's thesis, University of Amsterdam, 1999.

[55] M. Lipmann. *On-line Routing.* PhD thesis, Technische Universiteit Eindhoven, 2003.

[56] H. Liu, M. Queyranne, and D. Simchi-Levi. On the asymptotic optimality of algorithms for the flow shop problem with release dates. *Naval Research Logistics*, 2004. To appear.

[57] N. Megow and A. Schulz. On-line scheduling to minimize average completion time revisited. *Operations Research Letters*, 32:485–490, 2004.

[58] C. Phillips, C. Stein, and J. Wein. Minimizing average completion time in the presence of release dates. *Mathematical Programming*, 82:199–223, 1998.

[59] M. Pinedo. *Scheduling: Theory, Algorithms, and Systems.* Prentice Hall, second edition, 2002.

[60] H. Psaraftis, M. Solomon, T. Magnanti, and T. Kim. Routing and scheduling on a shoreline with release times. *Management Science*, 36(2):212–223, 1990.

[61] P. Raghavan. A statistical adversary for on-line algorithms. *Online Algorithms, DIMACS Series in Discrete Mathematics and Computer Science*, 7:79–83, 1992.

[62] M.W.P. Savelsbergh, R. Uma, and J. Wein. An experimental study of lp-based approximation algorithms for scheduling problems. *INFORMS Journal on Computing*, 17:123–136, 2005.

[63] A. Schulz and M. Skutella. The power of α-points in preemptive single machine scheduling. *Journal of Scheduling*, 5:121–133, 2002.

[64] A. Schulz and M. Skutella. Scheduling unrelated machines by randomized rounding. *SIAM Journal on Discrete Mathematics*, 15:450–469, 2002.

[65] S. Seiden. A guessing game and randomized online algorithms. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, pages 592–601, 2000.

[66] R. Sitters. *Complexity and approximation in routing and scheduling.* PhD thesis, Eindhoven University of Technology, 2004.

[67] D. Sleator and R. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.

[68] L. Stougie and A.P.A. Vestjens. Randomized algorithms for on-line scheduling problems: how low can't you go? *Operations Research Letters*, 30:89–96, 2002.

[69] J. Tsitsiklis. Special cases of traveling salesman and repairman problems with time windows. *Networks*, 22(3):263–282, 1992.

[70] A.P.A. Vestjens. *Online machine scheduling.* PhD thesis, Eindhoven University of Technology, 1997.

[71] J. Yang, P. Jaillet, and H. Mahmassani. Real-time multivehicle truckload pickup and delivery problems. *Transportation Science*, 38:135–148, 2004.

[72] N. Young. The k-server dual and loose competitiveness for paging. *Algorithmica*, 11(6):525–541, 1994.

Appendix A

A.1 Technical Details Concerning Equation (3.5)

In this section, we first discuss Equation (3.5):

$$\begin{aligned} & \ln\left(1+\frac{1}{m}-\frac{\gamma}{m}\right)+\frac{\gamma}{m} \\ = \; & \frac{e^{(-\gamma/m)}\left(1+\frac{1}{m}-\frac{\gamma}{m}-e^{(-\gamma/m)}\right)\left(me^{(\gamma/m)}-\gamma e^{(\gamma/m)}+\frac{1}{m}e^{(\gamma/m)}+1-m\right)}{1+\frac{1}{m}-\frac{\gamma}{m}} \end{aligned}$$

and we then discuss some details concerning δ_m and c_m.

A.1.1 Existence of $\gamma \in (0,1)$

We show that, for any finite m, there exists $\gamma \in (0,1)$ that satisfies Equation (3.5). Let

$$l(\gamma) = \ln\left(1+\frac{1}{m}-\frac{\gamma}{m}\right)+\frac{\gamma}{m}$$

and

$$r(\gamma) = \frac{e^{(-\gamma/m)}\left(1+\frac{1}{m}-\frac{\gamma}{m}-e^{(-\gamma/m)}\right)\left(me^{(\gamma/m)}-\gamma e^{(\gamma/m)}+\frac{1}{m}e^{(\gamma/m)}+1-m\right)}{1+\frac{1}{m}-\frac{\gamma}{m}}.$$

We have that $l(0) = \ln(1+\frac{1}{m})$ and $r(0) = \frac{1}{m}$; note that for finite m, $l(0) < r(0)$. For $\gamma = 1$, we have that $l(1) = \frac{1}{m}$ and $r(1) = e^{(-1/m)}\left(1-e^{(-1/m)}\right)\left((e^{(1/m)}-1)(m-1)+\frac{1}{m}e^{(1/m)}\right) \leq 0$, thus, $l(1) > r(1)$. As both $l(\gamma)$ and $r(\gamma)$ are continuous functions of γ, it is clear that there exists a value of $\gamma \in (0,1)$ such that $l(\gamma) = r(\gamma)$.

A.1.2 Uniqueness of $\gamma \in (0,1)$

Define $x = (1 + \frac{1}{m} - \frac{\gamma}{m})e^{(\gamma/m)} \geq 0$; Equation (3.5) can then be written as

$$\frac{x \ln x}{x-1} = m - \gamma + \frac{1}{m} - (m-1)e^{-(\gamma/m)}.$$

Let $l(\gamma) = \frac{x \ln x}{x-1}$ and $r(\gamma) = m - \gamma + \frac{1}{m} - (m-1)e^{-(\gamma/m)}$. Note that:

$$\begin{aligned} \frac{dl(\gamma)}{d\gamma} &= \frac{dl(\gamma)}{dx}\frac{dx}{d\gamma} = \frac{x - 1 - \ln x}{(x-1)^2}\frac{(1-\gamma)e^{(\gamma/m)}}{m^2} \geq 0 \\ \frac{dr(\gamma)}{d\gamma} &= -1 + \frac{m-1}{m}e^{-(\gamma/m)} < 0. \end{aligned}$$

Consequently, there is an unique solution to Equation (3.5).

A.1.3 Feasibility of $f(\alpha)$: $\delta_m \in (0,1),\ \forall m \geq 1$

Using the notation and analysis from above, we have that δ_m satisfies: $0 < ml(0) < \delta_m < ml(1) = 1$.

A.1.4 Calculation showing $\lim_{m\to\infty} \delta_m = 1$

Observing that $ml(0) = m \ln(1 + \frac{1}{m})$ and applying l'Hôpital's Rule,

$$\lim_{m\to\infty} m \ln(1 + \frac{1}{m}) = \lim_{m\to\infty} \frac{\left(\frac{1}{1+\frac{1}{m}}\right)\left(-\frac{1}{m^2}\right)}{-\frac{1}{m^2}} = 1.$$

A.1.5 Calculation showing $\lim_{m\to\infty} c_m = 1$

Note that $\gamma \in (0,1)$; we have that

$$\begin{aligned} \lim_{m\to\infty} c_m &= \lim_{m\to\infty} \frac{1/m}{e^{(\gamma/m)}(1 + \frac{1}{m} - \frac{\gamma}{m}) - 1} \\ &= \lim_{m\to\infty} \frac{-1/m^2}{e^{(\gamma/m)}(\frac{\gamma}{m^2} - \frac{1}{m^2}) + (1 + \frac{1}{m} - \frac{\gamma}{m})e^{(\gamma/m)}(-\frac{\gamma}{m^2})} \quad \text{(by l'Hôpital's Rule)} \\ &= 1. \end{aligned}$$

A.2 Technical Details for Chapter 5

The results in this section are by Vladimir Barzov.

Lemma 19: Almost Sure Convergence ([11]) *If* $\mathbb{E}[Y^r] < \infty$, $r \in \mathbb{N}$, *then* $\lim_{n\to\infty} \frac{Y_{(n)}}{n^\delta} = 0$ *almost surely, for any* $\delta \geq \frac{1}{r}$. *Moreover, the inequality does not always hold for* $0 < \delta < \frac{1}{r}$.

Proof If $A_n = \{|X_n| > \epsilon n^\delta\}$ and $B_m = \bigcup_{n\geq m} A_n$ then it is sufficient to prove

$$\lim_{m\to\infty} \mathbb{P}(B_m) = 0. \tag{A.1}$$

Notice that

$$\mathbb{P}(B_m) = 1 - \mathbb{P}(!B_m) = 1 - \mathbb{P}\left(\bigcap_{n\geq m} !A_n\right) = 1 - \prod_{n=m}^{\infty} \mathbb{P}\left(!A_n \middle| \bigcap_{i=m}^{n-1} !A_i\right) \tag{A.2}$$

Notice that for $n = m$ we have

$$\mathbb{P}\left(!A_n \middle| \bigcap_{i=m}^{n-1} !A_i\right) = \mathbb{P}(!A_m) = \mathbb{P}(X_m \leq \epsilon m^\delta) = \mathbb{P}\left(\bigcap_{i=1}^{m} Y_i \leq \epsilon m^\delta\right) =$$

$$= F(\epsilon m^\delta)^m, \tag{A.3}$$

where $F(y)$ is the distribution function of Y. Also, for $n > m$ we have

$$\mathbb{P}\left(!A_n \middle| \bigcap_{i=m}^{n-1} !A_i\right) = \mathbb{P}\left(X_n \leq \epsilon n^\delta \middle| \bigcap_{i=m}^{n-1} X_i \leq \epsilon i^\delta\right) = \mathbb{P}(Y_n \leq \epsilon n^\delta) =$$

$$= F(\epsilon n^\delta), \tag{A.4}$$

since $X_i \leq \epsilon i^\delta < \epsilon n^\delta$ and $X_i \leq X_n$ for $i < n$. Now, to prove (A.1), we use (A.2) where we plug in (A.3) and (A.4), so we want to show

$$\lim_{m\to\infty} \prod_{n=m}^{\infty} \mathbb{P}\left(!A_n \middle| \bigcap_{i=m}^{n-1} !A_i\right) = \lim_{m\to\infty} \left(F(\epsilon m^\delta)^m \prod_{n=m+1}^{\infty} F(\epsilon n^\delta)\right) = 1. \tag{A.5}$$

Let $\Delta(y) = 1 - F(Y)$, so $\Delta(y) \to 0$ as $y \to \infty$. Taking logarithms on both sides in (A.5), we are left to show

$$\lim_{m\to\infty} \left(m \ln(1 - \Delta(\epsilon m^{\delta})) + \sum_{n=m+1}^{\infty} \ln(1 - \Delta(\epsilon n^{\delta})) \right) = 0. \tag{A.6}$$

Since $\dfrac{\ln(1-\Delta)}{\Delta} \to -1$ as $\Delta \to 0$, for some Δ_0 and for all $0 \leq \Delta \leq \Delta_0$ we have

$$-2 \leq \frac{\ln(1-\Delta)}{\Delta},$$

and since $\Delta(\epsilon n^{\delta}) \to 0$ as $n \to \infty$ we have that for big enough m and all $n \geq m$ the following holds:

$$-2\Delta(\epsilon n^{\delta}) \leq \ln(1 - \Delta(\epsilon n^{\delta})) \leq 0.$$

Therefore, to prove (A.6), it is sufficient to show that

$$\lim_{m\to\infty} m\Delta(\epsilon m^{\delta}) = 0 \quad \text{and} \quad \lim_{m\to\infty} \sum_{n=m+1}^{\infty} \Delta(\epsilon n^{\delta}) = 0. \tag{A.7}$$

For all $y \geq 0$, denote $z = \left(\dfrac{y}{\epsilon}\right)^{\frac{1}{\delta}}$ and

$$g(z) = \Delta(\epsilon z^{\delta}) = \Delta(y). \tag{A.8}$$

We now use the fact that $\mathbb{E}[Y^r] < \infty$, and since $Y > 0$, it follows that for some constant C,

$$C \geq \int_0^{\infty} F'(y) y^r \mathrm{d}r = \int_0^{\infty} -\Delta'(y) y^r \mathrm{d}y =$$

$$= \int_0^{\infty} -\frac{\mathrm{d}\Delta(y)}{\mathrm{d}y} (\epsilon z^{\delta})^r \mathrm{d}y = \int_0^{\infty} -\frac{g'(z)\mathrm{d}z}{\mathrm{d}y} \epsilon^r z^{r\delta} \mathrm{d}y =$$

$$= \epsilon^r \int_0^{\infty} -g'(z) z^{r\delta} \mathrm{d}z. \tag{A.9}$$

For every n and from $r\delta \geq 1$, we have hence

$$C \geq \epsilon^r \int_0^\infty -g'(z) z^{r\delta} \mathrm{d}z \geq \epsilon^r \int_n^\infty -g'(z) z \mathrm{d}z \geq \epsilon^r n \int_n^\infty -g'(z) \mathrm{d}z = \epsilon^r n g(n),$$

so we set $M = C\epsilon^{-r}$ and have $M \geq ng(n)$ for all n.

Now, for any $z \in [n, n+1)$ and again from $r\delta \geq 1$ we have $z^{r\delta} \geq z \geq n$. Combined with (A.9), this gives us that for all n

$$C \geq \epsilon^r \sum_{k=0}^{n+1} k \int_k^{k+1} -g'(z) \mathrm{d}z = \epsilon^r \sum_{k=0}^{n} k(g(k) - g(k+1)) =$$

$$= \epsilon^r \left((g(1) - g(2)) + 2(g(2) - g(3)) + 3(g(3) - g(4)) + \cdots + n(g(n) - g(n+1)) \right) =$$

$$= \epsilon^r \left(g(1) + g(2) + g(3)) + \cdots + g(n) - ng(n+1) \right) \geq$$

$$\geq -\epsilon^r M + \epsilon^r \sum_{k=0}^{n} g(k), \tag{A.10}$$

since as we showed, $M \geq ng(n) \geq ng(n+1)$ for all n. This shows that the series $\sum_n g(n)$ is bounded by $M + \epsilon^{-r} C = 2M$ and therefore converges to some real number C_0. This has two consequences: First, we have that the partial sums of $\sum_n g(n)$ converge to C_0 and hence

$$\lim_{m \to \infty} \sum_{n=m+1}^{\infty} g(n) = 0,$$

which proves the second part of (A.7).

Second, from (A.10) we have also that the series $\sum_n n(g(n) - g(n+1))$ is bounded, hence convergent (all terms are positive) and so

$$\lim_{n \to \infty} \sum_{k=0}^{n} k(g(k) - g(k+1)) = C_2$$

for some constant C_2. Therefore,

$$ng(n+1) = \sum_{k=0}^{n} g(k) - \sum_{k=0}^{n} k(g(k) - g(k+1)) \to C_2 - C_1,$$

as $n \to \infty$. Obviously then $C_2 \geq C_1$. However, if it was the case that $C_2 > C_1$, then for large enough n, say $n \geq m$ we would have had

$$ng(n) \geq ng(n+1) > \frac{C_2 - C_1}{2} = \mu > 0,$$

so

$$C_1 = \sum_n g(n) \geq \sum_{n=0}^{m-1} + \sum_{n=m+1}^{\infty} \frac{\mu}{n} = \infty,$$

which is a contradiction. This shows that in fact $C_2 = C_1$ and so $ng(n+1)$ converges to 0. Therefore, $ng(n) = \frac{n}{n-1}(n-1)g(n) \to 1 \times 0 = 0$ as $n \to \infty$. This proves the first part of (A.7).

We now show that the restriction $\delta \geq \frac{1}{r}$ is in fact tight. Consider some $0 < \delta < \frac{1}{r}$ and take $\epsilon = 1$ for simplicity. If

$$F(y) = \begin{cases} 0, & y < 1 \\ 1 - \left(\frac{1}{y}\right)^{\frac{1}{\delta}}, & y \geq 1 \end{cases}.$$

This distribution satisfies $\mathbb{E}[Y^r] < \infty$ as

$$\int_{-\infty}^{\infty} F'(y) y^r \mathrm{d}r = \int_1^{\infty} \frac{1}{\delta} y^{-\frac{1}{\delta}-1} y^r \mathrm{d}y = \int_1^{\infty} \frac{1}{\delta} y^{r-\frac{1}{\delta}-1} \mathrm{d}y < \infty$$

as $r - \frac{1}{\delta} - 1 < -1$. However, as we will show, $\lim_{n\to\infty} \frac{Y_n}{n^\delta} > 0$, so it cannot be that $\lim_{n\to\infty} \frac{X_n}{n^\delta} = 0$. We have

$$P(Y_n < n^\delta) = F(n^\delta) = 1 - \frac{1}{n} = \frac{n-1}{n},$$

so

$$P\left(\bigcap_{n=m}^{\infty} (Y_n < n^\delta)\right) = \prod_{n \geq m} P(Y_n < n^\delta) = \frac{m-1}{m} \times \frac{m}{m+1} \times \frac{m+1}{m+2} \times \cdots = 0.$$

This shows that

$$\lim_{m\to\infty} P\left(\bigcap_{n=m}^{\infty}(Y_n < n^{\delta})\right) = 0 \neq 1,$$

so we do not have almost sure convergence of $\frac{Y_n}{n^{\delta}}$ to 0 and hence of $\frac{X_n}{n^{\delta}}$ to 0. The proof is complete. ■

Lemma 19: Convergence in Mean ([11]) *If* $\mathbb{E}[Y^r] < \infty$, $r \in \mathbb{N}$, *then* $\lim_{n\to\infty} \frac{Y_{(n)}}{n^{\delta}} = 0$ *in mean, for any* $\delta \geq \frac{1}{r}$. *Moreover, the inequality does not always hold for* $0 < \delta < \frac{1}{r}$.

Proof Let $F(y)$ be the cumulative distribution of Y. If there exists a number N such that $F(N) = 1$, then $0 \leq Y_n \leq N$ and so $0 \leq X_n \leq N$ for each n. We then have $\mathbb{E}[X_n^r] \leq N^r$, so clearly $\mathbb{E}\left[\left(\frac{X_n}{n^{\delta}}\right)^r\right] \to 0$ as $n \to \infty$.

Now, consider $F(y) < 1$ for all $y \geq 0$. We have that the cumulative distribution $G_n(x)$ of X_n is given by $F(x)^n$, as

$$\mathbb{P}(X \leq x) = \mathbb{P}\left(\bigcap_{i=1}^{n} Y_i \leq x\right) = \underbrace{F(x) \times F(x) \times \cdots F(x)}_{n-\text{times}}.$$

Therefore,

$$\mathbb{E}[X_n^r] = \int_0^{\infty} G'(x)x^r \mathrm{d}r = \int_0^{\infty} nF(x)^{n-1}F'(x)x^r \mathrm{d}x. \tag{A.11}$$

Now, let $\epsilon > 0$ be arbitrary. Since $\infty > \mathbb{E}[Y^r]$, for some constant M we have

$$M = \mathbb{E}[Y^r] = \int_0^{\infty} F'(y)y^r \mathrm{d}y.$$

Since this integral converges to M, there exists N such that

$$M \geq \int_0^{N} F'(y)y^r \mathrm{d}y \geq (1-\epsilon)M, \quad \int_N^{\infty} F'(y)y^r \mathrm{d}y \leq \epsilon M. \tag{A.12}$$

Now, since $F(N) = \gamma < 1$, for some large enough m we have $F(N)^m \leq \epsilon$ and therefore for all $n > m$ and $x \in [0, N]$ we have

$$\epsilon \geq F(N)^m \geq F(N)^{n-1} \geq F(x)^{n-1}. \tag{A.13}$$

Then, from (A.11), (A.13), and finally (A.12), we have

$$\mathbb{E}[X_n^r] = n\int_0^\infty F(x)^{n-1}F'(x)x^r\mathrm{d}x =$$

$$= n\int_0^N F(x)^{n-1}F'(x)x^r\mathrm{d}x + n\int_N^\infty F(x)^{n-1}F'(x)x^r\mathrm{d}x \leq$$

$$\leq n\int_0^N \epsilon F'(x)x^r\mathrm{d}x + n\int_N^\infty 1\times F'(x)x^r\mathrm{d}x = n\epsilon\int_0^N F'(x)x^r\mathrm{d}x + n\int_N^\infty F'(x)x^r\mathrm{d}x \leq$$

$$\leq n\epsilon M + n\epsilon M \leq 2nM\epsilon.$$

Therefore, from $\delta r \geq 1$, we have for all $n > m$ that

$$0 \leq E\left[\left(\frac{X_n}{n^\delta}\right)^r\right] = \frac{\mathbb{E}[X_n^r]}{n^{\delta r}} \leq \frac{2nM\epsilon}{n} \leq 2M\epsilon.$$

Since ϵ was an arbitrary positive number, it follows that

$$E\left[\left(\frac{X_n}{n^\delta}\right)^r\right] = 0.$$

We now show that the restriction $\delta \geq \frac{1}{r}$ is in fact tight. Consider some $0 < \delta < \frac{1}{r}$ and let

$$F(y) = \begin{cases} 0, & y < 1 \\ 1 - \left(\frac{1}{y}\right)^{\frac{1}{\delta}}, & y \geq 1 \end{cases}.$$

This distribution satisfies $\mathbb{E}[Y^r] < \infty$ because

$$\int_{-\infty}^\infty F'(y)y^r\mathrm{d}r = \int_1^\infty \frac{1}{\delta}y^{-\frac{1}{\delta}-1}y^r\mathrm{d}y = \int_1^\infty \frac{1}{\delta}y^{r-\frac{1}{\delta}-1}\mathrm{d}y < \infty$$

since $r - \frac{1}{\delta} - 1 < -1$. However, from (A.11) we have

$$\mathbb{E}[X_n^r] = \int_0^\infty nF(x)^{n-1}F'(x)x^r\mathrm{d}x = \int_0^\infty x^r\mathrm{d}F(x)^n.$$

We set $z = x^{\delta}, x \geq 1$, so $F(x) = 1 - \frac{1}{z}$ and therefore

$$\mathbb{E}[X_n^r] = \int_0^{\infty} z^{\delta r} \mathrm{d}\left(1 - \frac{1}{z}\right)^n = \int_0^{\infty} z^{\delta r} n \left(1 - \frac{1}{z}\right)^{n-1} \frac{1}{z^2} \mathrm{d}z =$$

$$= n \int_0^{\infty} \frac{(z-1)^{n-1}}{z^{n+1-\delta r}} \mathrm{d}z. \tag{A.14}$$

Now, since $\lim_{n\to\infty} \left(1 - \frac{1}{n}\right)^{n-1} = \frac{1}{e}$, for some m and all $n \geq m$ we have

$$\left(1 - \frac{1}{n}\right)^{n-1} > \frac{1}{2e}.$$

Therefore, for $z \geq n$ we have $\left(1 - \frac{1}{z}\right)^{n-1} \geq \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{1}{2e}$ and hence

$$(z-1)^{n-1} \geq \frac{z^{n-1}}{2e}. \tag{A.15}$$

Now, from (A.14) and (A.15) we get that for all $n \geq m$

$$\mathbb{E}[X_n^r] = n \int_0^{\infty} \frac{(z-1)^{n-1}}{z^{n+1-\delta r}} \mathrm{d}z \geq n \int_n^{\infty} \frac{(z-1)^{n-1}}{z^{n+1-\delta r}} \mathrm{d}z \geq$$

$$\geq \frac{n}{2e} \int_n^{\infty} \frac{z^{n-1}}{z^{n+1-\delta r}} \mathrm{d}z = \frac{n}{2e} \int_n^{\infty} z^{-2+\delta r} \mathrm{d}z = \frac{n}{2e} n^{-1+\delta r} = \frac{n^{\delta r}}{2e}.$$

Therefore, for all $n \geq m$ holds

$$E\left[\left(\frac{X_n}{n^{\delta}}\right)^r\right] = \frac{\mathbb{E}[X_n^r]}{n^{\delta r}} \geq \frac{\frac{1}{2e} n^{\delta r}}{n^{\delta r}} = \frac{1}{2e},$$

so $\lim_{n\to\infty} E\left[\left(\frac{X_n}{n^{\delta}}\right)^r\right] \neq 0$. ■

Lemma 24 ([11]) *If* $\mathbb{E}[X^4] < \infty$, *then for any* $k \geq 0$,

$$\lim_{n\to\infty} \frac{1^k X_1 + 2^k X_2 + 3^k X_3 + \cdots + n^k X_n}{n^{k+1}} = \frac{\mu_X}{k+1},$$

almost surely.

Proof Obviously, we can only consider X with mean zero, because the transformation $X - \mu$ will preserve both the fourth moment condition and what we have to prove [1]. So we want to show that

$$U_n = \frac{1^k X_1 + 2^k X_2 + \cdots + n^k X_n}{n^{k+1}} \rightarrow 0 \quad \text{a.s.},$$

where $\mathbb{E}[U_n] = 0$. We now state the following lemmas:

Lemma 26 *There exists a constant c, such that for all n*

$$\mathbb{E}[U_n^4] \leq \frac{c}{n^2}.$$

Lemma 27 *If* $P_n(\epsilon) = \mathbb{P}(|B_n - B| > \epsilon)$ *satisfies* $\sum P_n(\epsilon) < \infty$ *for all* $\epsilon > 0$, *then*

$$B_n \rightarrow B$$

almost surely.

The proof of Lemma 26 follows and Lemma 27 is a standard result.

Proof of Lemma 26: Since $\mathbb{E}[X_i] = 0$ and X_i, X_j are independent for $i \neq j$, we can see that only the fourth powers X_i^4 and the coupled squares $X_i^2 X_j^2$ will survive the operator $\mathbb{E}$ after removing the brackets of

$$\mathbb{E}[U_n^4] = E\left[\left(\frac{1^k X_1 + \cdots + n^k X_n}{n^{k+1}}\right)^4\right] =$$

$$= \frac{1}{n^{4k+4}}(1^{4k} + 2^{4k} + \cdots + n^{4k})\mathbb{E}[X^4] + \frac{1}{n^{4k+4}}\left(\sum_{i \neq j} i^{2k} j^{2k}\right)\mathbb{E}[X^2]^2$$

$$< \frac{1}{n^{4k+4}}(n^{4k} \times n)\mathbb{E}[X^4] + \frac{1}{n^{4k+4}}(n^2 \times n^{2k} n^{2k})\mathbb{E}[X^2]^2 = \frac{\mathbb{E}[X^4]}{n^3} + \frac{\mathbb{E}[X^2]^2}{n^2} < \frac{c}{n^2}$$

[1] We use the fact here that $\lim_{n \rightarrow \infty} \frac{1^k + 2^k + \cdots + n^k}{n^{k+1}} = \frac{1}{k+1}$

for *all* n and some big enough constant c. The proof of Lemma 26 is complete.

From Markov's inequality applied to $U_n^4 \geq 0$ and from Lemma 26, we have

$$P_n = \mathbb{P}(|U_n| > \epsilon) = \mathbb{P}(U_n^4 > \epsilon^4) \leq \frac{\mathbb{E}[U_n^4]}{\epsilon^4} \leq \frac{c}{\epsilon^4 n^2} = \frac{c_1}{n^2}$$

for some constant $c_1 = \frac{c}{\epsilon^4}$. Therefore,

$$\sum_{n=1}^{\infty} P_n \leq c_1 \sum_{n=1}^{\infty} \frac{1}{n^2} < \infty,$$

for some constant c_2. Therefore, applying Lemma 27 with $B_n = U_n, B = 0$, we get $U_n \to 0$ as desired. ■

We now prove a lemma that is needed in the proof of Lemma 25.

Lemma 28 ([11]) *Let X be a distribution with $\mathbb{E}[X^4] < \infty$. If $X_n \sim X, n = 1, 2, \ldots$ are i.i.d., then*

$$\lim_{n\to\infty} \frac{nX_1 + (n-1)X_2 + X_3 + \cdots + X_n}{n^2} = \frac{\mathbb{E}[X]}{2}, \quad \text{a.s.}$$

Proof Notice that

$$\frac{nX_1 + (n-1)X_2 + \cdots + X_n}{n^2} = \frac{(n+1)}{n}\frac{X_1 + X_2 + \cdots + X_n}{n} - \frac{X_1 + 2X_2 + \cdots + nX_n}{n^2}.$$

The first sum converges to $\mathbb{E}[X]$ from the law of large numbers, and the second one to $\frac{\mathbb{E}[X]}{2}$ from Lemma A.2 applied with $k = 1$. The difference converges to $\mathbb{E}[X] - \frac{\mathbb{E}[X]}{2} = \frac{\mathbb{E}[X]}{2}$. ■

Lemma 25 ([11]) *Let $\{X_i\}$ and $\{Y_j\}$ be two sequences of i.i.d. random variables. If $\mathbb{E}[X^2] < \infty$ and $\mathbb{E}[Y^2] < \infty$, then*

$$\lim_{n\to\infty} \frac{\sum_{j=1}^{n} X_j \sum_{i=1}^{j} Y_i}{n^2} = \frac{\mathbb{E}[X]\mathbb{E}[Y]}{2}, \quad a.s.$$

Proof Let $\mathbb{E}[X] = \mu, \mathbb{E}[Y] = \nu$, both of which exist for obvious reasons. Define

$$C_n = \frac{Y_1X_1 + (Y_1 + Y_2)X_2 + \cdots + (Y_1 + \cdots + Y_n)X_n}{n^2} = U_n + V_n + W_n,$$

where

$$U_n = \frac{\sum_{i=1}^{n}\left((X_i - \mu)\sum_{j=1}^{i}(Y_j - \nu)\right)}{n^2},$$

$$V_n = \nu\frac{(X_1 - \mu) + 2(X_2 - \mu) + \cdots + n(X_n - \mu)}{n^2},$$

and

$$W_n = \mu\frac{nY_1 + (n-1)Y_2 + \cdots + Y_n}{n^2}.$$

We can directly apply Lemma A.2 (with $k = 1$) to show $V_n \to 0$ a.s. since the variable in question, $X - \mu$, satisfies $\mathbb{E}[(X - \mu)^4] < \infty$ and $\mathbb{E}[X - \mu] = 0$. Next, from Lemma 28, we have $W_n \to \mu\frac{\nu}{2}$ almost surely. To prove that $\lim_{n\to\infty} C_n = \frac{\mu\nu}{2}$, it only remains to show $\lim_{n\to\infty} U_n = 0$. For simplicity of notation, consider $\mathbb{E}[X] = \mu = 0, \mathbb{E}[Y] = \nu = 0$, so that

$$U_n = \frac{Y_1X_1 + (Y_1 + Y_2)X_2 + \cdots + (Y_1 + \cdots + Y_n)X_n}{n^2}.$$

Notice that $\mathbb{E}[U_n] = 0$ as Y_i and X_j are independent. Next, if we denote $S_k = \sum_{i=1}^{k} Y_i$, we have

$$\mathbb{E}[(n^2U_n)^4] = E\left[S_1^4X_1^4 + S_2^4X_2^4 + \cdots + S_n^4X_n^4 + 6\sum_{1\le i<j\le n} S_i^2S_j^2X_i^2X_j^2\right] =$$

$$\mathbb{E}[X^4]\sum_{i=1}^{n}\mathbb{E}[S_i^4] + 6\mathbb{E}[X^2]^2\sum_{1\le i<j\le n} E\left[S_i^2S_j^2\right], \tag{A.16}$$

as all X_i are independent and so only the even powers of X_i would remain. Next,

$$\mathbb{E}[S_k^4] = \sum_{i=1}^{k}\mathbb{E}[Y_i^4] + 6\sum_{1\le i\le j<k}\mathbb{E}[Y_i^2]\mathbb{E}[Y_j^2] =$$

$$k\mathbb{E}[Y^4] + 6\mathbb{E}[Y^2]^2\frac{k(k+1)}{2} < c_1k^2 \tag{A.17}$$

for some constant c_1, so for the first part of (A.16) we have

$$\mathbb{E}[X^4]\sum_{i=1}^{n}\mathbb{E}[S_i^4] < \mathbb{E}[X^4]\sum_{i=1}^{n} c_1 n^2 < c_2 n^3$$

for some constant c_2. Next, if $1 \leq i < j \leq n$, denote $\Delta_{ij} = S_j - S_i$. Notice that S_i and Δ_{ij} are independent and mean zero. Then

$$\mathbb{E}[S_i^2 S_j^2] = \mathbb{E}[S_i^2(S_i^2 + 2S_i\Delta_{ij} + \Delta_{ij}^2)] =$$

$$= \mathbb{E}[S_i^4] + 2\mathbb{E}[S_i^3\Delta_{ij}] + \mathbb{E}[S_i^2\Delta_{ij}^2] =$$

$$= \mathbb{E}[S_i^4] + 2\mathbb{E}[S_i^3]\mathbb{E}[\Delta_{ij}] + \mathbb{E}[S_i^2]\mathbb{E}[T_{ij}^2] =$$

$$= \mathbb{E}[S_i^4] + 0 + i(j-i)\mathbb{E}[Y^2]^2 < c_1 i^2 + n^2\mathbb{E}[Y^2]^2 < c_3 n^2$$

for some constant c_3. Therefore, the second sum in (A.16) satisfies

$$6\mathbb{E}[X^2]^2 \sum_{1\leq i<j\leq n} E\left[S_i^2 S_j^2\right] < 6\mathbb{E}[X^2]^2 \binom{n}{2} c_3 n^2 < c_4 n^4$$

for some constant c_4. Now, from (A.16), we have

$$\mathbb{E}[U_n^4] \leq \frac{c_4 n^4}{n^8} = \frac{c_4}{n^4}.$$

Therefore, from Markov's inequality,

$$P_n = \mathbb{P}(|U_n| > \epsilon) = \mathbb{P}(U_n^4 > \epsilon^4) \leq \frac{\mathbb{E}[U_n^4]}{\epsilon^4} \leq \frac{c_4}{\epsilon^4 n^4}.$$

This shows that

$$\sum_{n=1}^{\infty} P_n < \infty$$

and hence from Lemma 27, applied to $B_n = U_n$, $B = 0$, we have $U_n \to 0$ a.s. ■