DYNAMIC ONLINE RESOURCE ALLOCATION PROBLEMS

BY

GE YU

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Doctoral Committee:

Professor Sheldon H. Jacobson, Chair
Professor Rayadurgam Srikant
Professor Chandra Chekuri
Professor Bruce Hajek

ABSTRACT

Online resource allocation problems consider assigning a limited number of available resources to sequentially arriving requests with the objective to maximize rewards. With the emergence of e-business, applications such as online order fulfilment and customer service require real-time resource allocation decisions to guarantee high service quality and customer satisfaction. Other typical applications include operation room scheduling, organ transplant, and passenger screening in aviation security. This dissertation approaches the dynamic online resource allocation problem by considering two models: multi-objective sequential stochastic assignment problems and online interval scheduling problems.

Multi-objective sequential stochastic assignment problems are a class of matching problems. A fixed number of jobs arrive sequentially to be assigned to one of the available workers, with an n-dimensional value vector revealed upon each arrival. The objective is to maximize the reward vector given by the product of the job value vector and worker's success rate. We conduct a complete asymptotic analysis for three classes of Pareto optimal policies, with convergence rates and asymptotic objective values provided.

Online interval scheduling problems consider reusable resources, where an adversarial sequence of jobs with fixed lengths are to be assigned on available machines. The objective is to maximize the total reward for completed jobs given by the product of the job value and the machine weight. For homogeneous machines, we propose a Pairing-m algorithm, which is 2-competitive for even m and $(2 + 2/m)$-competitive for odd m. For heterogeneous machines, two classes of approximation algorithms, Cooperative Greedy algorithms and Prioritized Greedy algorithms, are compared using competitive ratios with respect to varying machine weight ratios. We also provide lower bounds for competitive ratios of deterministic online scheduling algorithms in various scenarios.

Stochastic online interval scheduling problems consider a sequence of jobs drawn from a given distribution. For identically and independently distributed jobs with a known distribution, we propose 2-competitive online algorithms for both equal-length and memoryless-length jobs. For job sequences with a random order of arrivals, we propose e-competitive and $e^2/(e-1)$-competitive online algorithms for both equal-length and memoryless-length jobs. We further extend these results to jobs with a random order of arrivals and geometric arrivals with parameter p.

We propose a primal-dual analysis framework for online interval scheduling algorithms for both adversarial and stochastic job sequences. We formulate the online interval scheduling as a linear program with a corresponding dual program. For stochastic job sequences, we use complementary slackness conditions and weak duality to derive optimal algorithms and upper bounds for the optimal reward, respectively. For adversarial sequences, we use weak duality to compute the competitive ratios of scheduling algorithms.

To my family, for their love and support.

ACKNOWLEDGMENTS

First, I would like to thank my advisor, Prof. Sheldon H Jacobson, for his wise advice and patient mentoring throughout my PhD. His deep insights, broad knowledge, rigorous research style and great writing fashion have not only benefited me tremendously during my pursuit for PhD, but will also shape my research and work style in the future. I indeed enjoy working with him and sincerely appreciate all his guidance.

I would like to thank my committee members, Prof. Chandra Chekuri, Prof. Bruce Hajek, and Prof. Rayadurgam Srikant, for their valuable feedback and helpful suggestions. I thank Prof. Chandra Chekuri for introducing me to the broad research area of online interval scheduling problems and approximation algorithms. I thank Prof. Bruce Hajek for teaching me random processes and all the insightful discussions on game theory. I have taken three courses with Prof. Rayadurgam Srikant, and I thank him for the great teaching and showing me the beautiful diversity in electrical and computer engineering.

I would like to thank Prof. Negar Kiyavash for leading me through my master's degree. I am thankful for all the scientific training and kind help she has offered me.

Life in graduate school can be tough sometimes, and I am grateful to have supportive teammates whenever I need them. I would like to thank my group members Arash Khatibi and Hee Youn Kwon for sharing their intelligence and offering their friendship. I would like to thank Shaileshh Bojja Venkatakrishnan for all the discussions on homework and research ideas. I would like to thank all my labmates for their selfless help and creating a stimulating environment to work in.

I am thankful to all my friends and relatives, who have helped me settle down into my new life in a foreign country and make me feel not alone.

Last but not least, I would like to thank my amazing parents for their

unconditional love and support. Looking back at the path I have come through, I realize how much they have sacrificed to make my PhD journey possible. I owe all I have earned to their believing in me, encouraging me and being there for me. I would like to thank Mr. Matthew Potok, for his love, support and patience. I have lost count of how many times he proofread my paper drafts, rehearsed as the first audience for my presentations, and offered constructive suggestions. I am blessed to have him by my side through my most difficult times in graduate school.

This research has been supported in part by the Air Force Office of Scientific Research under Grant No. FA9550-15-1-0100. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the United States Government, or the Air Force Office of Scientific Research.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Online resource allocation problems have been studied in operations research, which covers a wide variety of applications, such as asset selling in economics [1], organ transplant in medical research [2], Adwords bidding in online auctions [3], and aviation screening in homeland security [4]. With different assumptions and objective functions, the resource allocation problem can be formulated into different mathematical models. For example, if the objective is to minimize the span time and balance machine loads, the resulting model becomes a job shop scheduling problem; if the objective is to maximize the number of matched one-to-one pairs, the resulting model becomes a bipartite matching problem; if the objective is to maximize the profit from assigning resources to requests, the resulting model becomes a sequential stochastic assignment problem.

Our research is motivated by the aviation security screening problem. The Transportation Security Administration (TSA) is promoting a risk-based screening strategy to improve both air travel security and passenger experience [5]. Under the risk-based screening strategy, passengers are assigned to different levels of screening procedures based on their risk values. Different levels of screening procedures employ different devices and security personnel, and hence have different rates of true alarm (a higher screening level employs devices with more enhanced imaging and detecting technologies and has a higher rate of true alarm). Therefore, the capacity of screening devices in each level can be seen as a kind of heterogeneous resource, where each screening level possesses a *weight*, with a natural interpretation as the conditional probability of sending out a warning signal if a passenger with a threat is assigned to this level. How to allocate the limited resources of screening devices to arriving passengers in a most efficient way has been the main concern for policy-makers.

Besides this normal security screening procedure, the airport screening

system is also challenged by more complicated tasks. Take the *enhanced entry screening* for the 2014 Ebola Hemorrhagic Fever (or simply, Ebola) outbreak as an example. Under this enhanced screening, all passengers coming from or transferring through three West African countries (Guinea, Liberia, and Sierra Leone) are required to be routed to one of five main airports to undergo *risk assessment*, and may be subject to a 21-day quarantine requirement based on their risk factors [6]. An effective screening assignment policy that uses the available resources efficiently is an essential component to prevent the spread of Ebola.

These two screening problems have been studied in the literature. [4] formulates the aviation security problem as a sequential stochastic assignment problem with the objective of maximizing the total *security reward*. However, a single objective is generally not enough. For example, the policy-maker may prefer a policy that maximizes expected security and expected confidence value at the same time. [7] formulates the Ebola screening problem as a multi-objective sequential stochastic assignment problem (MOSSAP) to improve the process for managing screening and monitoring assignments. The objective function components consist of maximizing the expected number of passengers correctly assigned to each category and minimizing the expected number of social contacts to be covered for mistakenly assigned passengers. Their results are based on experimental study and hence do not provide any theoretical guarantee. Moreover, these models may be too limiting since they are built on the assumption of indefinite occupation of a service capacity (resource) and hence do not take the real-time reusability of resources into consideration.

Our research focuses on two mathematical models that fall into the category of online resource allocation problems: multi-objective sequential stochastic assignment problems (MOSSAP) and interval scheduling problems. For MOSSAP, we provide a complete asymptotic analysis to help policy-makers compare different Pareto optimal policies. For the interval scheduling model, we consider reusable resources and take the dynamic changes of resources into consideration. We study online interval scheduling problems for both adversarial and stochastic job sequences, with a primal-dual analysis framework provided. Our results can be applied to applications ranging from call management in customer service centers to resource management in cloud computing platforms.

1.1 Multi-objective Sequential Stochastic Assignment Problems

Multi-objective sequential stochastic assignment problems combine two research topics: multi-objective optimization problems and sequential stochastic assignment problems. Typical applications include: (a) Airport security screening assignments, where a policy-maker prefers a policy that maximizes the total expected security and confidence level simultaneously. (b) Customer service management, where a coordinator prefers an algorithm that maximizes the number of served customers and the satisfaction of customers simultaneously. (c) Online investment decisions, where an investor prefers a strategy that maximizes the expected profit and minimizes the expected cost simultaneously.

[8] introduces the sequential stochastic assignment problem (SSAP), where T workers with known success rates $p_1 \leq p_2 \leq \ldots \leq p_T$ are to be assigned to T sequentially arriving tasks. Each task value $\bar{C}_t$ (random variables) is revealed upon arrival, for $t = 1, 2, \ldots, T$. The objective is to maximize the total expected reward $\mathbb{E}[\sum_{t=1}^{T} p_{j_t} \bar{C}_t]$, where j_t is the index of the worker assigned to perform the t^{th} task with value $\bar{C}_t$. The task values are assumed to be independent and identically distributed (IID) random variables with a known cumulative distribution function (cdf). An optimal policy is obtained based on recursive equations to compute threshold values for each task assignment.

Variations of SSAP have been widely studied. [9] studies the SSAP with random arrival times and discounted rewards under different arrival distributions and discount functions. [10] studies the SSAP with a random number of arriving tasks in two cases: the distribution of the number of tasks has finite or infinite support. [11] studies SSAP with task value distributions not necessarily independent. [12] studies the SSAP with the distributions of two successive task values governed by a known Markov chain. [13] relaxes the assumptions in [12] and studies the SSAP in a partially observable Markov chain. SSAP with multi-item assignments and vector offers has been discussed in [14–16]. The SSAP application in aviation security screening has appeared in [4, 5, 17, 18]. The SSAP application in organ transplant has appeared in [2, 19]. These above mentioned works all focus on the SSAP with a single objective function, while we consider a more general and complicated

class of SSAP with multiple objectives.

Another research topic related to SSAP is the generalized online assignment problem in the *uniform arrival model* (i.e., the arrival order is completely random). Adwords problems [20], resource allocation problems [21,22] and online matching problems [23,24] fall into this category. In such a setting, a sequence of requests arrive online, with a profit that can only be gained if fulfilled. Resources have certain capacities, and the objective is to maximize the total expected profits from fulfilled requests subject to resource capacities. Requests are not assumed to follow any distribution, but are assumed to be picked by an adversary beforehand and have a uniform arrival order. No optimal algorithm is known for this uniform arrival model in literature, but approximation algorithms that guarantee a fraction of the underlying optimal reward have been widely studied. MOSSAP uses a stronger assumption that task arrivals are IID with given distributions to obtain an optimal policy.

Asymptotic analysis for SSAP with a single objective has also attracted research interest. [25] studies the limiting performance of the SSAP optimal policy and computes the asymptotic expected reward per task as the number of tasks approaches infinity. [26], [27] study the limiting performance of the SSAP optimal policy with multiple assignment categories and propose asymptotically optimal policies with a fixed number of thresholds for the two following scenarios: (a) task value distribution is known and is IID, (b) task value distributions are unknown but governed by a known ergodic Markov chain. All the previous asymptotic analyses focus on the asymptotic expected reward value, and none of them has provided a convergence rate. Moreover, the task values are assumed to follow continuous distributions in the above-mentioned literature. The asymptotic analysis for MOSSAP uses techniques similar to those in [25] and [26], but relaxes the continuity restriction on the cdf of the task value. We provide asymptotic expected rewards and convergence rates for multiple objective functions for trade-off analysis between different Pareto optimal policies.

Several methods have been proposed to solve multi-criteria optimization problems, for both the on-line and off-line settings. For recent progress in this area, see [28] and [29]. A well-known approach for solving multi-criteria problems is the weighted sum method [30], where multiple objective functions are summed up into a single objective function weighted by a vector. Although

this method is simple and works for the on-line optimization setting, there are pitfalls to consider before using it: (a) the weighted sum method usually does not guarantee the generation of the complete set of Pareto optimal solutions; (b) the weight vector may not be easily specified beforehand by the decision maker. Another widely applied method is the ϵ-constrained method, where only one objective is kept as the main optimization objective at a time while the others are transformed into constraints specified by the ϵ-vector [31]. Though the ϵ-constrained method works as well as the weighted sum method in terms of generating Pareto optimal policies in the off-line setting, it is difficult to introduce this method into the online optimization setting and obtain closed-form expressions for multiple objective function values. Moreover, the values of the ϵ-vector remain hard to specify beforehand.

1.1.1 Our results

Our primary contribution is a complete asymptotic analysis for the general class of MOSSAP with product-form vector rewards and discrete task value distributions. The objective of MOSSAP is to maximize each component of the n-dimensional vector of the expected reward per task. We start from the case where all workers are homogeneous and focus on *Pareto optimal policies* for MOSSAP. The set of Pareto optimal policies for MOSSAP is generated by the weighted sum method and the SSAP optimal policy. The asymptotic expected reward per task for each component of the reward vector under Pareto optimal policies is provided. Three different classes of policies are considered and proved to be asymptotically Pareto optimal, with convergence rates provided for comparison. We also show how to extend results for homogeneous workers to heterogeneous workers.

1.2 Online Interval Scheduling Problems

For online interval scheduling problems, there are sequentially arriving jobs to be scheduled on a single machine or multiple machines. Each job has the following characteristics: (a) an arrival time; (b) a length, the amount of time required to completer a job; (c) a deadline; and (d) a value, the reward for completing a job. We consider the case where the deadline of a job is

equal to the sum of the arrival time and the length of the job, and hence each job can be represented by an interval along the time axis and must be scheduled immediately upon arrival.

The offline interval scheduling problem on non-identical machines is NP-complete [32]. Other results for the offline interval scheduling problem can be found in [33], [34], [35], [36], and [37]. For the online scheduling problem for equal-value jobs, [38] proposes a optimal Greedy algorithm, GOL. The general online interval scheduling problem for arbitrary-value jobs does not have any approximation algorithms with finite worst case guarantees [39]. Additional assumptions are needed to obtain algorithms with finite competitive ratios. Such assumptions include job sequences with equal lengths and arbitrary values [40], job sequences with values uniformly proportional to lengths [41], job sequences with monotone deadlines and values [42], and job sequences with a deterministic relationship between values and lengths [39]. For a detailed summary and comparison of these results, see [43].

Another widely used technique to construct online scheduling algorithms is *randomization*. [44] first proposes a randomized $(2+\sqrt{3})$-competitive algorithm for scheduling C-benevolent jobs on a single machine. [42] proposes a 3-competitive algorithm on a single machine for *monotone instances*, where the order of right points of job intervals coincides with the order of left points of jobs and job values are non-decreasing. [45] proposes 2-competitive barely random algorithms for equal-length and C-benevolent job sequences, respectively.

The problem of scheduling on multiple machines has been extensively studied over the years. [41] proposes a cooperative 2-competitive algorithm on two identical machines for jobs with values uniformly proportional to lengths. [40] proposes a 3.5822-competitive algorithm on two machines for jobs with equal lengths and arbitrary values. They provide a lower bound of 4/3 (2) for scheduling C-benevolent jobs on multiple (two) machines. As for scheduling on more than two machines, [46] proposes a 4-competitive Greedy algorithm, ALG, for scheduling C-benevolent jobs on multiple *uniformly related machines* (i.e., each machine has a service speed). [47] proposes a 2(2+2/(2m-1))-competitive algorithm for scheduling equal-length jobs on even (odd) number of machines.

All variations of the online interval scheduling problem reported in the literature focus on the objective of maximizing the total value of completed

jobs, where machines have no weights; hence, these approaches are homogeneous in terms of rewards [34, 37, 46, 48, 49]. This problem formulation fails to capture some real-life applications, such as assigning jobs to workers with different success rates, where success rate is defined as the probability of completing an assigned job. The expected reward for an assignment is hence the product of the job value and the worker's success rate. For example, in the aviation screening problem, each passenger with a risk value is treated as a job with a value and a length (the amount of time needed for screening). Each screening level is treated as a machine for executing jobs with different *weights* (different levels of screening procedures employ different devices and security personnel, and hence, have different rates of true alarms). The weights of screening levels have a natural interpretation as the conditional probability of sending out a warning signal if a passenger with a threat is assigned to this level. The objective of maximizing the total reward of completed job assignments (passenger screening assignments) is hence given by the product of the value of jobs and the weight of their assigned machines.

Existing results for online interval scheduling problems focus on the *worst-case* analysis, which considers adversarial job arrivals and is known to be overly pessimistic. We consider stochastic online interval scheduling problems, RSSAP, where sequentially arriving jobs are drawn from a given distribution. We evaluate the average performance of an algorithm with respect to all possible job sequences, and hence our approach differs from existing works on online interval scheduling problems. RSSAP has similarities with M/M/s/N queuing systems, where s servers (machines) and a finite buffer of size $N - s$ are available for arriving jobs [50]. The job arrivals in this queuing system are assumed to follow a Poisson process, with IID exponential inter-arrival times. The service times of servers are assumed to follow an exponential distribution. The objective of an optimization problem for a queuing system may be to minimize the steady-state sojourn time (i.e., service and waiting time) of jobs, maximize the throughput within a fixed time interval, or minimize the probability of losing jobs. There are significant differences between the general M/M/s/N queuing system and RSSAP: (1) there is no queuing space in RSSAP, and assignments are made immediately upon each arrival; (2) the time required to complete a job in RSSAP is given and fixed; (3) the objective of RSSAP is to maximize the total reward of completed jobs, given by the product of job values and machine weights.

Online bipartite matching problems and online budgeted bidding problems with IID arrivals are related to RSSAP. [23, 51] consider an online stochastic matching problem with IID arrivals. They propose approximation algorithms using disjoint optimal offline matchings. [52] considers an online bipartite matching problem with unknown IID arrivals and shows that the RANKING algorithm achieves a competitive ratio of 1.532. [24, 53] consider an online stochastic matching problem in the *random arrival order* setting, which is more general than the IID setting. [53] proposes an e-competitive algorithm for edge-weighted matching problems. [24] uses a family of factor-revealing linear programs to show the RANKING algorithm is 1.437-competitive for unweighted matching. However, these results are not trivially generalizable to the stochastic interval scheduling problem since bipartite matching problems assume indefinite occupation of resources. That is, once a resource and a request are matched, the matched resource will not be available anymore. The online budgeted bidding problem allows multiple matchings to a single resource, yet they do not consider the real-time reusability of resources. That is, a resource can be allocated to subsequent requests as long as the budget allows, which is not feasible for the online interval scheduling problem. The most relevant work to RSSAP is [54], which considers a generalization of the secretary problem with equal-length temporary employment. They assume the value of the secretary follows a random arrival order and the arrival time of the secretary follows a known distribution, which is different from our settings.

Primal-dual techniques are commonly used for deriving approximation algorithms for online resource allocation problems [55]. [56] uses primal-dual analysis to formulate an online learning algorithm for discounted Markov decision processes (MDP) with unknown transition probabilities and transition costs. [57] applies the primal-dual analysis in solving weighted online paging problems. [58] proposes near-optimal algorithms for online resource allocation problems under a random order of arrivals, where the constraint matrix as well as the corresponding objective coefficient is revealed column by column.

Primal-dual techniques can also be used for analyzing algorithms. [59] uses the primal-dual technique to give a simpler proof for the e-competitiveness of the optimal threshold policy for secretary problems. [60] uses similar techniques to analyze the RANKING algorithm [61] for online bipartite matching

problems, and provides a simpler proof for the $(1-1/e)$-competitiveness. [62] studies the Bellman equation of a MDP, concluding that the dual value of the policy function is the optimal value for both infinite-horizon with discounted reward and finite-horizon MDPs. [63] considers the JISPk scheduling problem, where the algorithm has to choose an interval from a tuple of k feasible intervals for each job to maximize the total number of scheduled jobs. They evaluate the approximability of the LP relaxation for the original problem using weak duality. None of these papers use primal-dual techniques to develop or analyze the performance of an algorithm for online interval scheduling problems. We introduce an approach using the primal-dual technique to solve both stochastic and adversarial online interval scheduling problems.

1.2.1 Our results

We study a few variations of the online interval scheduling problem and provide different analysis techniques. Specifically, we consider equal-length jobs and C-benevolent jobs, which capture many applications (see [39], [46] and [45]).

Chapter 3 considers scheduling a sequence of C-benevolent jobs on multiple homogeneous machines, generalizing the problem proposed by [39] on a single machine. For two machines, we propose a 2-competitive Cooperative Greedy algorithm. We further generalize the algorithm to multiple machines and propose a Pairing-m algorithm, which is deterministic 2-competitive for even number of machines and randomized (2+2/m)-competitive for odd number of machines. The Pairing-m algorithm improves the 4-competitive algorithm given by [46]. We provide lower bounds of 2 and 1.436 for the competitive ratio of any deterministic online scheduling algorithms on two and three machines, respectively. Therefore, the Cooperative Greedy algorithm achieves the best possible competitive ratio for scheduling C-benevolent jobs on two machines.

Chapter 4 extends results in Chapter 3 to multiple weighted machines and focuses on two classes of online algorithms: Cooperative Greedy algorithms and Prioritized Greedy algorithms, with competitive ratios provided. We show that when the weight ratios between machines are small, the *Cooperative Greedy* algorithm outperforms the *Prioritized Greedy* algorithm. As

the weight ratios increase, the *Prioritized Greedy* algorithm outperforms the *Cooperative Greedy* algorithm. Moreover, as the weight ratios approach infinity, the competitive ratio of the *Prioritized Greedy* algorithm approaches four. We also provide lower bounds of 3/2 and 9/7 for the competitive ratio of any deterministic online scheduling algorithm on two and three weighted machines, respectively, which hold for arbitrary machine weights.

Chapter 5 introduces and analyzes stochastic online interval scheduling problems, where workers are assigned to perform a job for a certain amount of time and then return to be reassigned for future arriving jobs. The job assignment is assumed to be irrevocable and non-preemptive, which means once a job is assigned, the job must be completed by the worker without any suspension or change. The objective is to maximize the total expected reward for completed jobs, which is given by the product of the job value and the success rate of the assigned worker. We consider three kinds of job arrivals: (1) IID job arrivals with a given job value distribution, (2) job arrivals following a random order, and (3) job arrivals following a random order and a geometric arrival with parameter p. Approximation algorithms for both cases are proposed, with competitive ratios provided. For each case, we consider two classes of job sequences: (a) equal-length jobs and (b) *memoryless-length jobs*.

Chapter 6 provides a primal-dual approach for analyzing algorithms for both stochastic and adversarial online interval scheduling problems. We formulate the online interval scheduling problem as a general linear program and then give the corresponding dual program for each specific case. The linear program for the online interval scheduling problem is different from existing works, since the constraints for the primal linear program are constantly changing due to the scheduling and completing of jobs, and the coefficient for the objective function is not given a priori. Therefore, feasible solutions to the primal and dual programs have to dynamically adapt to these changes. For stochastic online interval scheduling problems, we propose an optimal randomized algorithm for scheduling equal-length arbitrary-value jobs on a single machine using strong duality. We also provide an upper bound for the optimal reward for scheduling equal-length arbitrary-value jobs on multiple machines using weak duality. For adversarial online interval scheduling problems, we consider scheduling special kinds of jobs, *C-benevolent* jobs, on a single machine using three different algorithms. We use weak duality to ana-

lyze each algorithm by constructing a feasible solution to the dual program, matching known competitive ratios.

CHAPTER 2

ASYMPTOTIC ANALYSIS FOR MOSSAP

This chapter provides an asymptotic analysis of multi-objective sequential stochastic assignment problems (MOSSAP). In MOSSAP, a fixed number of tasks arrive sequentially, with an n-dimensional value vector revealed upon arrival. Each task is assigned to one of a group of known workers immediately upon arrival, with the reward given by an n-dimensional product-form vector. The objective is to maximize each component of the expected reward vector. We provide expressions for the asymptotic expected reward per task for each component of the reward vector, under all Pareto optimal policies. We propose another two classes of asymptotically Pareto optimal policies, with one class preserving the optimal convergence rate and the other requiring little computational effort. We also study the convergence rates of these three classes of Pareto optimal policies for MOSSAP. These convergence rates also apply to the classic single-objective sequential stochastic assignment problem with discrete task value distributions.

2.1 Formulation

Consider T tasks to be sequentially assigned to $\eta \leq T$ workers irrevocably with $\eta \in \mathbb{Z}^+$. For each task, an n-dimensional value vector is revealed upon arrival. Denote the value vector for the t^{th} task by the vector of random variables $\boldsymbol{\mathcal{A}}(t) \triangleq (\mathcal{A}_1(t), \ldots, \mathcal{A}_n(t))$, with the random variable $\mathcal{A}_j(t)$ defined as the j^{th} component of the task value vector, for $j = 1, 2, \ldots, n$ and $t = 1, 2, \ldots, T$. We do not assume that the components of $\boldsymbol{\mathcal{A}}(t)$ are independent of each other. However, we assume $\mathcal{A}_j(t)$ to be discrete and $\boldsymbol{\mathcal{A}}(t)$ to be IID across tasks. Denote the marginal probability mass functions (pmf) for $\mathcal{A}_j(t)$ by $p_{\mathcal{A}_j}(\alpha_j)$ for $j = 1, 2, \ldots, n$, and the joint pmf by $p_{\boldsymbol{\mathcal{A}}}(\boldsymbol{\alpha})$ (i.e., the pmf of $\boldsymbol{\mathcal{A}}(t)$). Denote the realized value vector of the t^{th} task by

$\boldsymbol{\alpha}(t) \triangleq (\alpha_1(t), \ldots, \alpha_n(t))$. We assume η homogeneous workers. Denote the success rates of T workers by $\{\tau_i\}_{i=1}^T$, and set $\tau_i = 1$ for $i = 1, 2, \ldots, \eta$ and $\tau_i = 0$ for $i = \eta+1, \eta+2, \ldots, T$, where we create $T - \eta$ virtual workers with success rate zero for simplicity in description (without specific explanation, "worker" refers to the original worker). Let i_t denote the index of the worker assigned to the t^{th} task. The number of workers is referred to as the *capacity*. Define the *complementary capacity ratio* as one minus the ratio of the capacity to the total number of tasks, denoted by $\zeta = 1 - \frac{\eta}{T}$.

A policy for MOSSAP defines a sequence of task assignments. Let the binary random variable $X_t^\Phi \in \{0, 1\}$ denote the t^{th} task assignment under policy Φ: $X_t^\Phi = 1(0)$ denotes assigning the t^{th} task to a worker ($\tau_{i_t} = 1(0)$). Policy Φ may be *pure* or *mixed*. A mixed policy Φ consists of a sequence of random variables, denoted by $\mathcal{P}^\Phi \triangleq \{\mathcal{X}_t^\Phi\}_{t=1}^T$, with $\mathcal{X}_t^\Phi$ defined as the conditional probability of assigning the t^{th} task to a worker, given the task value vectors that have been revealed. If a policy is pure, then task assignments are *deterministic* given a sequence of task value vectors.

The objectives are to maximize the expected reward per task for each component of the reward vector. The expectations are taken with respect to the distributions of the sequence of task value vectors and the randomness of the policy assignments (for mixed policies only). Denote the expected reward per task for the j^{th} component of the reward vector under policy Φ by $r_j(\Phi)$, for $j = 1, 2, \ldots, n$, and let $\mathbf{r}(\Phi) = (r_1(\Phi), r_2(\Phi), \ldots, r_n(\Phi))$. Since MOSSAP has n objective functions that typically do not admit the same optimal policy, we aim to generate *Pareto optimal* policies for MOSSAP. Denote the set of admissible policies for MOSSAP by Ψ^η, referred to as the *feasible region*. More precisely, the on-line policy $\Phi \in \Psi^\eta$ attempts to optimize, for $j = 1, 2, \ldots, n$,

$$r_j(\Phi) \triangleq \frac{1}{T}\mathbb{E}[\sum_{t=1}^{T} X_t^\Phi \mathcal{A}_j(t)], \tag{2.1}$$

where

$$\Psi^\eta \triangleq \{\Phi : \sum_{t=1}^{T} X_t^\Phi = \eta\}.$$

Definition 1 gives the formal definition of Pareto optimal policies for MOSSAP.

Definition 1. *A policy $\Phi \in \Psi^\eta$ is said to be Pareto optimal for MOSSAP if there does not exist another policy $\Phi' \in \Psi^\eta$ such that $r_j(\Phi) \leq r_j(\Phi')$ for all*

$j = 1, 2, \ldots, n$, *with at least one strict inequality.*

For multi-objective optimization problems, Pareto optimal policies are typically not unique.

2.2 Pareto Optimal Policies for MOSSAP

The set of Pareto optimal policies for MOSSAP can be obtained using the weighted sum method. Let $\mathbf{w} = (w_1, w_2, \ldots, w_n)$ denote the *non-negative weight vector* for the objective functions of MOSSAP, with $w_j \geq 0$ for $j = 1, \ldots, n$ and $\sum_{j=1}^{n} w_j > 0$ (with abuse of notation, denote this by $\mathbf{w} \geq 0$). Using the weighted sum method, $r_j(\Phi),\ j = 1, \ldots, n$ are combined into a single weighted objective function, denoted by $R_w(\Phi)$:

$$\begin{aligned} R_w(\Phi) = \sum_{j=1}^{n} w_j r_j(\Phi) &= \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T} X_t^{\Phi}\left(\sum_{j=1}^{n} w_j \mathcal{A}_j(t)\right)\right] \\ &= \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T} X_t^{\Phi}\left(\sum_{j=1}^{n} \mathcal{G}(t)\right)\right], \end{aligned} \tag{2.2}$$

where the random variable $\mathcal{G}(t)$ denotes the one-dimensional *combined value* for the t^{th} task,

$$\mathcal{G}(t) \equiv \sum_{j=1}^{n} w_j \mathcal{A}_j(t). \tag{2.3}$$

The maximization of $R_w(\Phi)$ over the feasible region Ψ^{η} is referred to as the *weighted objective sequential assignment problem (WOSA) indexed by* $\mathbf{w}$ (or simply, WOSA-$\mathbf{w}$). If only one of the weight vector components is non-zero (i.e., $w_j > 0$ and $w_{j'} = 0$ for $j' \neq j$), then $\max_{\Phi \in \Psi^{\eta}} R_w(\Phi)$ reduces to $\max_{\Phi \in \Psi^{\eta}} r_j(\Phi)$. If there exists some $\mathbf{w} \geq 0$ such that policy $\Phi \in \Psi^{\eta}$ maximizes the objective function $R_w(\Phi)$ defined in (2.2), then Φ is said to be *optimal* for WOSA.

Note that the weighted sum method in general does not guarantee a bijection between Pareto optimal policies for MOSSAP and optimal policies for WOSA. Since neither redundant nor omitted policies are desired, if we are to benefit from the single objective optimization using WOSA, an extra *pruning* step is needed to exclude redundant policies from the set of optimal policies for WOSA. Moreover, convexity of the feasible region Ψ^{η} and affinity of $r_j(\Phi)$

for $j = 1, \ldots, n$ guarantee that no Pareto optimal policies are omitted using the weighted sum method.

If policy $\Phi \in \Psi^{\eta}$ maximizes the objective function $R_w(\Phi)$ (2.2) for some $\mathbf{w} > 0$ (i.e., $w_j > 0$ for all j), then Φ is Pareto optimal for MOSSAP from *Theorem 3.1.2* [64, p. 78]. Therefore, the pruning for Pareto optimal policies is only needed if policy $\Phi \in \Psi^{\eta}$ maximizes $R_w(\Phi)$ for some $\mathbf{w}$ with zero-components. [65] discusses a straightforward pruning method using a definition of $\mathbb{M}$-optimal policies for WOSA when there are only two objective functions (i.e., $n = 2$). For the general case with $n > 2$, that definition cannot be directly applied. Instead, we propose a *brute-force* pruning algorithm using the asymptotic results provided in Section 2.3. For brute-force pruning, values of the n objective functions under all optimal policies for WOSA are enumerated for comparison. This is further discussed in Section 2.7.

To obtain the property of convexity, consider the set of mixed policies in the feasible region Ψ^{η} (pure policies are considered as a special kind of mixed policy, and hence, included in the set of mixed policies). We extend the feasible region to $\Psi^{\eta+} \triangleq \{\Phi : \mathbb{E}[\sum_{t=1}^{T} X_t^{\Phi}] = \eta\}$, where the expectation is taken with respect to X_t^{Φ} (i.e., $\mathbb{E}[\sum_{t=1}^{T} X_t^{\Phi}] = \eta$ holds for any sequence of task value vectors). We only use $\Psi^{\eta+}$ given its convexity. Since optimal policies for WOSA are all pure policies and $\{\Phi_p : \Phi_p \text{ is pure and } \Phi_p \in \Psi^{\eta}\} = \{\Phi_p : \Phi_p \text{ is pure and } \Phi_p \in \Psi^{\eta+}\}$ [65], maximizing each $r_j(\Phi)$ over Ψ^{η} and $\Psi^{\eta+}$ are equivalent. Moreover, all admissible mixed policies in $\Psi^{\eta+}$ define a convex set [65], denoted by

$$\Xi^{\eta} \triangleq \Big\{\mathcal{P}^{\Phi} = \{\mathcal{X}_t^{\Phi}\}_{t=1}^{T} : \sum_{t=1}^{T} \mathcal{X}_t^{\Phi} = \eta, \text{ for } \mathcal{X}_t^{\Phi} \in [0,1]\Big\}.$$

Proposition 1 generalizes *Proposition 3* in [65], and its proof is similar so is omitted.

Proposition 1. *The objective functions $r_j(\Phi)$ for $j = 1, 2, \ldots, n$ of MOSSAP are all affine functions of $\Phi \in \Psi^{\eta+}$.*

Given convexity of $\Psi^{\eta+}$ and Proposition 1, all Pareto optimal policies for MOSSAP can be generated using the weighted sum method for all $\mathbf{w} \geq 0$ from *Theorem 3.1.4* [64, p. 79].

2.2.1 Optimal policies for WOSA

The optimal policy for WOSA-**w** can be generated by applying *Theorem 1* [8], trimmed especially for discrete task value distributions. The objective of WOSA-**w** is to maximize the expected weighted reward per task for T task assignments, and the reward for assigning a task to a worker is $\mathcal{G}(t)$.

Denote the cdf for $\mathcal{G}(t)$ by $F_{\mathcal{G}}(\gamma)$, where γ is the realized combined value defined as $\gamma_t \equiv \sum_{j=1}^{n} w_j \alpha_j(t)$. Since $\mathcal{A}_j(t)$ are discrete for all j, $\mathcal{G}(t)$ is also discrete. Denote these discrete values by $0 < G_1 < G_2 < \ldots < G_L$ ($G_0 = 0$ and $F_{\mathcal{G}}(G_L) = 1$). Applying the law of total probability, the pmf $p_{\mathcal{G}}(\gamma)$ for $\mathcal{G}(t)$ is

$$p_{\mathcal{G}}(\gamma) = \sum_{\{\boldsymbol{\alpha}:\sum_{j=1}^{n} w_j \alpha_j = \gamma\}} p_{\boldsymbol{\mathcal{A}}}(\boldsymbol{\alpha}). \tag{2.4}$$

Let $\eta(t)$ denote the *remaining capacity* before the t^{th} task assignment for $t = 0, 1, 2, \ldots, T$, with $\eta(0) = \eta(1) = \eta$ ($t = 0$ has no task arrival and describes the *initial stage* with T tasks to be assigned). For the t^{th} task assignment, there exist threshold values

$$-\infty = a_{0,t} \leq a_{1,t} \leq \ldots \leq a_{T-t+1,t} = +\infty, \tag{2.5}$$

obtained using the recursive equations given by [8] with the integral substituted by a summation,

$$a_{i,t} = \left(\sum_{\gamma=g^u_{i-1,t+1}}^{g^l_{i,t+1}} \gamma p_{\mathcal{G}}(\gamma) \right) + a_{i-1,t+1} F_{\mathcal{G}}(a_{i-1,t+1}) + a_{i,t+1} \left(1 - F_{\mathcal{G}}(a_{i,t+1})\right), \tag{2.6}$$

where

$$g^u_{i-1,t+1} \equiv \min_{G_{l'} \in \{G_1, G_2, \ldots, G_L\}} G_{l'} > a_{i-1,t+1}, \qquad g^l_{i,t+1} \equiv \max_{G_{l'} \in \{G_1, G_2, \ldots, G_L\}} G_{l'} \leq a_{i,t+1}, \tag{2.7}$$

for $i = 1, 2, \ldots, T - t$ and $t = 0, 1, 2, \ldots, T$. If $\mathcal{G}(t) \in (a_{i-1,t}, a_{i,t}]$ for some $i \geq T - t - \eta(t) + 2$, then the t^{th} task is assigned to a worker. Specifically,

this policy (Φ1), referred to as the *SSAP optimal policy*, is given by

$$X_t^{\Phi 1} = \begin{cases} 1, & \text{if } \mathcal{G}(t) > a_{T-t-\eta(t)+1,t}, \\ 0 & \text{otherwise,} \end{cases} \qquad (\Phi 1)$$
$$\eta(t+1) = \eta(t) - X_t^{\Phi 1}, \qquad t = 1, 2, \ldots, T.$$

Theorem 1 is given without proof. For proof, refer to proofs of *Theorem 1* [8] or *Theorem 6* [65].

Theorem 1. *Policy* (Φ1) *with threshold values defined by (2.6) is optimal for WOSA-***w***, when* T *tasks are to be assigned. Moreover, the threshold values in the initial stage,* $\{a_{i,0}\}_{i=1}^{T}$*, are the expected combined values for the* T *tasks.*

To see the computational effort for policy (Φ1), note that since $\mathcal{A}_j(t)$ are discrete random variables for $j = 1, 2, \ldots, n$, $\boldsymbol{\mathcal{A}}(t)$ can only take on a value from a finite set, $\{A_1^1, A_1^2, \ldots, A_1^{K_1}\} \times \ldots \times \{A_n^1, A_n^2, \ldots, A_n^{K_n}\}$, with the cardinality bounded above by $\Pi_{j=1}^{n} K_j$. Since the combined value of each task can only assume one of the L values $\{G_1, G_2, \ldots, G_L\}$ ($L \leq \Pi_{j=1}^{n} K_j$), the time complexity to compute each threshold value in (2.6) for policy (Φ1) is $O(L) = O(\Pi_{j=1}^{n} K_j)$. For the t^{th} passenger out of T passengers, $T-t$ threshold values are required. Therefore, the total time complexity is $O(T^2 \Pi_{j=1}^{n} K_j)$ and the space requirement is $O(T^2)$.

Next, we compute the objective function values for WOSA-**w** and MOSSAP under policy (Φ1). First, we introduce some notations and definitions. Denote the i^{th} smallest combined value of T tasks to be assigned by the random variable $\hat{\mathcal{G}}_T^{(i)}$. Then, $\mathbb{E}[\hat{\mathcal{G}}_T^{(i)}] = a_{i,0}$, for $i = 1, 2, \ldots, T$ with $\{a_{i,0}\}_{i=1}^{T}$ defined by (2.6). Denote the j^{th} component of the task value vector that results in the i^{th} smallest combined value of T tasks ($\hat{\mathcal{G}}_T^{(i)}$), by the random variable $\hat{\mathcal{A}}_T^{(j)(i)}$, with the subscript indicating the total number of tasks to be assigned. Define $b_{i,t}^j \triangleq \mathbb{E}[\hat{\mathcal{A}}_{T-t}^{(j)(i)}]$, for $i = 1, 2, \ldots, T-t$ and $t = 0, 1, \ldots, T-1$. Therefore, $b_{i,t}^j$ is the expected value of the j^{th} component of the task value vector that results in the i^{th} smallest combined value of $T-t$ tasks. Specifically, if $t = 0$, then

$$b_{i,0}^j = \mathbb{E}[\hat{\mathcal{A}}_T^{(j)(i)}] = \mathbb{E}\left[\mathbb{E}[\hat{\mathcal{A}}_T^{(j)(i)} | \hat{\mathcal{G}}_T^{(i)}]\right], \qquad (2.8)$$

and $\{b_{i,0}^j\}_{i=1}^{T}$ are the expected values of the j^{th} component of the task value vector of T tasks to be assigned.

Corollary 1 provides the objective function values for WOSA-**w** and MOSSAP under policy ($\Phi 1$) without proof. For proof, refer to proofs of *Corollaries 1 and 2* [65].

Corollary 1. *The objective function values for WOSA-***w** *and MOSSAP under policy* ($\Phi 1$) *are*

$$R_w(\Phi 1) = \max_{\Phi \in \Psi^\eta} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{T} X_t^{\Phi} \mathcal{G}(t)] = \frac{1}{T} \sum_{i=T-\eta+1}^{T} a_{i,0}, \tag{2.9}$$

and

$$r_j(\Phi 1) = \frac{1}{T} \mathbb{E}[\sum_{t=1}^{T} X_t^{\Phi 1} \mathcal{A}_j(t)] = \frac{1}{T} \sum_{i=T-\eta+1}^{T} b_{i,0}^j, \tag{2.10}$$

for $j = 1, 2, \ldots, n$. *Here,* $\{a_{i,t}\}$ *are defined by (2.6) and* $\{b_{i,t}^j\}$ *defined by* (2.8) *are given by the recursive equations*

$$b_{i,t}^j = (\sum_{\gamma'=g_{i-1,t+1}^u}^{g_{i,t+1}^l} \mathbb{E}[\mathcal{A}_j(t) | \mathcal{G}(t) = \gamma'] p_{\mathcal{G}}(\gamma')) + b_{i-1,t+1}^j F_{\mathcal{G}}(a_{i-1,t+1}) + b_{i,t+1}^j (1 - F_{\mathcal{G}}(a_{i,t+1})),$$

for $i = 1, 2, \ldots, T-t$ *and* $t = 0, 1, 2, \ldots, T$, *with* $b_{0,t}^j = 0$, $b_{T-t+1,t}^j = A_j^{K_j}$, *and* $g_{i-1,t+1}^u$, $g_{i,t+1}^l$ *defined by (2.7).*

2.3 Asymptotic Analysis under the SSAP Optimal Policy

In this section, we present the asymptotic expected rewards per task for WOSA-**w** and MOSSAP under the SSAP optimal policy as the total number of tasks T approaches infinity. Moreover, we compute the limits of threshold values for the SSAP optimal policy ($\Phi 1$) and show that successive threshold values collapse to the jump points of the cdf of the combined task value, $F_{\mathcal{G}}(\gamma)$. The threshold collapse occurs in SSAP with discrete task value distributions, which results in reduction of computational effort for the SSAP optimal policy when T is sufficiently large, as discussed in Section 2.4.

We assume the complementary capacity ratio ζ to be fixed as T approaches infinity, which means that η increases proportionally with T (i.e., $\eta = \lfloor T(1-\zeta) \rfloor$, where $\lfloor \cdot \rfloor$ denoting the floor function, $\lfloor x \rfloor = \max_{N \in \mathbb{Z}} N \leq x$).

In the following, first, we present the asymptotic expected weighted reward per task for WOSA-**w**. Then, these asymptotic analysis results are used to compute the asymptotic expected reward per task for each component of the reward vector for MOSSAP.

2.3.1 Asymptotic analysis for WOSA-w

Denote the optimal asymptotic expected weighted reward per task for $R_w(\Phi)$ (2.2) by $\rho_w^\zeta(\mathbf{w})$. Then, $\rho_w^\zeta(\mathbf{w})$ is achieved under the SSAP optimal policy ($\Phi 1$) from Theorem 1. Theorem 2 provides a closed-form expression for $\rho_w^\zeta(\mathbf{w})$.

Theorem 2. *The optimal asymptotic expected weighted reward per task for WOSA-***w** *is*

$$\rho_w^\zeta(\mathbf{w}) = \left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k) \right) + q G_{l+1} p_{\mathcal{G}}(G_{l+1}), \tag{2.11}$$

where

$$q = \frac{F_{\mathcal{G}}(G_{l+1}) - \zeta}{p_{\mathcal{G}}(G_{l+1})}, \tag{2.12}$$

and $F_{\mathcal{G}}(G_l) \leq \zeta < F_{\mathcal{G}}(G_{l+1})$ *for some* $l \in \{0, 1, \ldots, L-1\}$.

Proof: See Appendix A.

From Theorem 2, both the weight vector **w** and the complementary capacity ratio ζ influence the optimal asymptotic expected weighted reward per task $\rho_w^\zeta(\mathbf{w})$. To see this, the distribution of $\mathcal{G}(t)$ depends on the weight vector **w** (by (2.4)), and q is determined by the distribution of $\mathcal{G}(t)$ and the complementary capacity ratio ζ.

Next, we show the threshold values in the initial stage collapse on the jump points of the cdf $F_{\mathcal{G}}(\gamma)$ using Theorem 2 (see Figure 2.1). This is a special property for SSAP with discrete task value distributions, which leads to further discussions on reducing computational efforts for the SSAP optimal policy. Moreover, the limit values of these thresholds are essential to the asymptotic analysis for MOSSAP. Corollary 2 provides the limit of threshold value in the initial stage, $a_{\lceil T\theta \rceil, 0}$, for a fixed $0 < \theta < 1$ as $T \to +\infty$, which is interpreted as the $\lceil T\theta \rceil^{th}$ smallest expected combined value of T tasks to be assigned (with $\lceil \cdot \rceil$ denoting the ceiling function, $\lceil x \rceil = \min_{N \in \mathbb{Z}} N \geq x$).

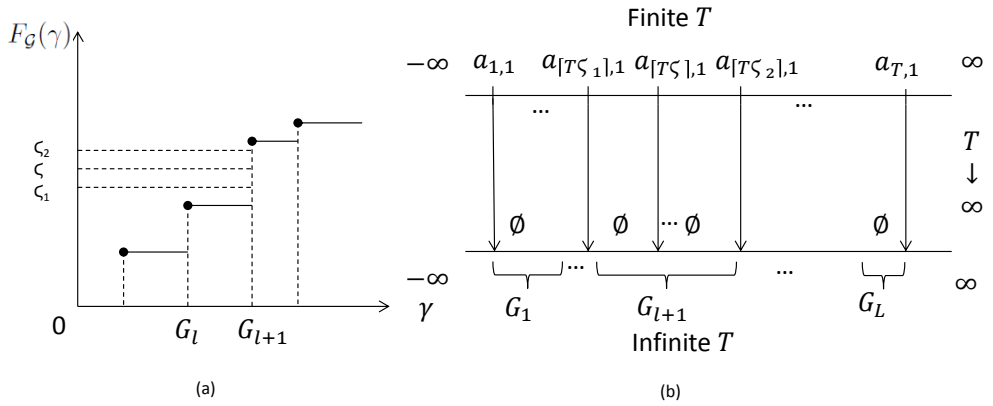

Figure 2.1: Convergence collapse of threshold values for the SSAP optimal policy with the discrete distribution $F_{\mathcal{G}}(\gamma)$ as $T \to +\infty$.

Corollary 2. *The limit of the threshold value in the initial stage defined by (2.6) is*

$$\lim_{T\to+\infty} a_{\lceil T\theta\rceil,0} = G_{l+1},$$

for $F_{\mathcal{G}}(G_l) < \theta < F_{\mathcal{G}}(G_{l+1})$ *and* $l = 0, 1, \ldots, L-1$.

Proof: See Appendix A.

Corollary 2 cannot be applied if $\theta = F_{\mathcal{G}}(G_l)$ for $l = 1, \ldots, L-1$. This can be seen by following the proof of Corollary 2, where the limits of the lower and upper bounds, $G_l \leq \lim_{T\to+\infty} \inf a_{\lceil T\theta\rceil,0}$ and $\lim_{T\to+\infty} \sup a_{\lceil T\theta\rceil,0} \leq G_{l+1}$, will not be equal. However, since the values of $\{a_{\lceil TF_{\mathcal{G}}(G_l)\rceil,0}\}_{l=1}^{L}$ are uniformly bounded above by G_L (by definition (2.6)), their exact values will not influence the following asymptotic analysis for MOSSAP.

2.3.2 Asymptotic analysis for MOSSAP

Theorem 2 provided a closed-form expression for the asymptotic expected weighted reward per task for the objective function $R_w(\Phi)$ of WOSA-**w** under the SSAP optimal policy (Φ1). Recall that MOSSAP has n objective functions. Moreover, an optimal policy that maximizes the weighted objective function $R_w(\Phi)$ does not necessarily maximize $r_j(\Phi)$ for all $j = 1, 2, \ldots, n$ at the same time. Therefore, $\rho_w^{\zeta}(\mathbf{w})$ alone is not sufficient to evaluate the performance of a policy for MOSSAP. We provide expressions for asymptotic

expected rewards per task for the n objective functions of MOSSAP under policy ($\Phi 1$), which capture how weight vectors influence the n-dimensional reward vector.

Consider the j^{th} component of the expected reward per task under the SSAP optimal policy ($\Phi 1$), $r_j(\Phi 1)$ (2.1), for $j = 1, 2, \ldots, n$. First, we show the uniform convergence of $\{b^j_{i,0}\}^T_{i=1}$ as $T \to +\infty$, which follows from the uniform convergence of $\{a_{i,0}\}^T_{i=1}$ as $T \to +\infty$. Then, the limit value of $r_j(\Phi 1)$ (not necessarily optimal for $r_j(\Phi)$) is computed using (2.10) by summing up $\{b^j_{i,0}\}^T_{i=\lceil T\zeta\rceil+1}$ and taking the limit as $T \to +\infty$ (an interchange of summation and limit is needed).

Lemma 1 states the uniform convergence of $b^j_{\lceil T\theta\rceil,0}$ for θ in a compact interval. This guarantees the interchangeability of summation and limit as $T \to +\infty$ in (2.10).

Lemma 1. *For any $\epsilon_1 > 0,\ \epsilon_2 > 0$, and any $l = 0, 1, \ldots, T-1$, define the compact interval $I^l_{\epsilon_1,\epsilon_2} \triangleq [F_{\mathcal{G}}(G_l) + \epsilon_1, F_{\mathcal{G}}(G_{l+1}) - \epsilon_2]$. Then, $b^j_{\lceil T\theta\rceil,0}$ defined by (2.8) converges uniformly as*

$$\lim_{T\to+\infty} b^j_{\lceil T\theta\rceil,0} = \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{l+1}],$$

for $\theta \in I^l_{\epsilon_1,\epsilon_2}$ as $T \to +\infty$, for $j = 1, 2, \ldots, n$.

Proof: See Appendix A.

Theorem 3 provides expressions for the asymptotic expected rewards per task for MOSSAP under policy ($\Phi 1$).

Theorem 3. *The asymptotic expected reward per task for $r_j(\Phi 1)$ (2.1) of MOSSAP is*

$$\lim_{T\to+\infty} r_j(\Phi 1) = \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) > G_{l+1}]\mathbb{P}_{\mathcal{G}}(\mathcal{G}(t) > G_{l+1}) + q\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{l+1}]p_{\mathcal{G}}(G_{l+1}), \tag{2.13}$$

with q defined by (2.12), where $F_{\mathcal{G}}(G_l) \le \zeta < F_{\mathcal{G}}(G_{l+1})$ for some $l \in \{0, 1, \ldots, L-1\}$ and $j = 1, 2, \ldots, n$.

Proof: See Appendix A.

Let $\rho^\zeta_j(\mathbf{w})$ denote the asymptotic expected reward per task for $r_j(\Phi 1)$ under the policy ($\Phi 1$), for $j = 1, 2, \ldots, n$. Therefore,

$$\rho^\zeta_j(\mathbf{w}) \triangleq \lim_{T\to+\infty} r_j(\Phi 1), \text{ for } j = 1, 2, \ldots, n. \tag{2.14}$$

2.4 An SSAP Mixed Policy

In this section, we propose an *SSAP mixed policy*, which achieves asymptotic optimality for WOSA-**w**, motivated by the threshold value collapse for policy (Φ1) from Corollary 2. The SSAP mixed policy improves by constant in computational effort compared with policy (Φ1), without impairing the convergence rate (see Section 2.6). We show that each component of the asymptotic expected reward per task for MOSSAP is the same under the SSAP mixed policy and policy (Φ1).

If $F_{\mathcal{G}}(G_l) < \zeta < F_{\mathcal{G}}(G_{l+1})$ for some $l \in \{0, 1, \ldots, L-1\}$, define

$$\nu_\zeta \triangleq \min\left(\frac{\zeta - F_{\mathcal{G}}(G_l)}{1 - F_{\mathcal{G}}(G_l)}, \frac{F_{\mathcal{G}}(G_{l+1}) - \zeta}{F_{\mathcal{G}}(G_{l+1})}\right). \tag{2.15}$$

Theorem 4 provides an SSAP mixed policy, which is based on the threshold values defined in (2.6) and makes the first ν_ζ fraction of task assignments based on the same single threshold.

Theorem 4. *Suppose policy* (Φ2) *assigns the* T *tasks as follows:*

$$X_t^{\Phi 2} = \begin{cases} 1, & \text{if } \mathcal{G}(t) > G_{l+1}, \\ 1, & \text{with probability } q \text{ if } \mathcal{G}(t) = G_{l+1}, \\ 0, & \text{otherwise, for } t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor, \end{cases} \tag{Φ2}$$

$$X_t^{\Phi 2} = X_t^{\Phi 1}, \text{ for } t = \lfloor \nu_\zeta T \rfloor + 1, \lfloor \nu_\zeta T \rfloor + 2, \ldots, T,$$

with q *defined by (2.12). Then, policy* (Φ2) *achieves the optimal asymptotic expected reward per task for WOSA-***w***, where* $F_{\mathcal{G}}(G_l) \leq \zeta < F_{\mathcal{G}}(G_{l+1})$ *for some* $l \in \{0, 1, \ldots, L-1\}$.

Proof: See Appendix A.

The computational effort for policy (Φ2) is the same as the computational effort for policy (Φ1) with $T - \lfloor \nu_\zeta T \rfloor$ tasks and hence is $O((1-\nu_\zeta)^2 T^2 \Pi_{j=1}^n K_j)$ in time and $O((1-\nu_\zeta)^2 T^2)$ in space.

Theorem 5. *The asymptotic value for* $r_j(\Phi 2)$ *(2.1) of MOSSAP is*

$$\lim_{T \to +\infty} r_j(\Phi 2) = \rho_j^\zeta(\mathbf{w}),$$

for $j = 1, 2, \ldots, n$, *with* $\rho_j^\zeta(\mathbf{w})$ *given by* (2.13) *and* (2.14).

Proof: See Appendix A.

2.5 A Single-Threshold Mixed Policy

In this section, we propose a *single-threshold mixed policy*, which achieves asymptotic optimality for WOSA-**w**. This policy requires little computational effort but has a slower convergence rate (see Section 2.6). We show that each component of the asymptotic expected reward per task for MOSSAP is the same under the single-threshold mixed policy and policy (Φ1).

Theorem 6. *Suppose a single-threshold mixed policy* (Φ3) *assigns tasks as follows:*

$$X_t^{\Phi 3} = \begin{cases} 1, & \text{if } \mathcal{G}(t) > G_{l+1},\ \eta(t) > 0, \\ 1, & \text{with probability } q \text{ if } \mathcal{G}(t) = G_{l+1},\ \eta(t) > 0, \\ 0, & \text{otherwise,} \end{cases} \tag{Φ3}$$

$$\eta(t+1) = \eta(t) - X_t^{\Phi 3} \text{ and } \eta(1) = \lfloor T(1-\zeta) \rfloor, \tag{2.16}$$

for $t = 1, 2, \ldots, T$ with q given by (2.12). Then, policy (Φ3) *achieves the optimal asymptotic expected reward per task for WOSA-**w**, where $F_{\mathcal{G}}(G_l) \leq \zeta < F_{\mathcal{G}}(G_{l+1})$ for some $l \in \{0, 1, \ldots, L-1\}$.*

Proof: See Appendix A.

Theorem 7. *The asymptotic value for $r_j(\Phi 3)$ (2.1) of MOSSAP is*

$$\lim_{T \to +\infty} r_j(\Phi 3) = \rho_j^{\zeta}(\mathbf{w}),$$

for $j = 1, 2, \ldots, n$, with $\rho_j^{\zeta}(\mathbf{w})$ given by (2.13) *and* (2.14).

Proof: See Appendix A.

2.6 Convergence Rate Analysis

This section provides convergence rates for the three aforementioned policies, i.e., the SSAP optimal policy (Φ1), the SSAP mixed policy (Φ2), and

the single-threshold mixed policy ($\Phi 3$). For the SSAP optimal policy, we show that threshold values defined by (2.6) converge to their limits with an exponential rate using properties of order statistics. Then we prove the convergence rate of the expected weighted reward per task $R_w(\Phi 1)$ as $O(1/\sqrt{T})$ using these exponential convergence rates. This convergence rate, referred to as the *optimal convergence rate*, applies to a general class of SSAP with a single objective function, as long as the task value has finite discrete support and the assignment rewards are of the product-form. Moreover, we prove the convergence rate of each component of the expected reward per task $r_j(\Phi 1)$ is $O(1/\sqrt{T})$ for $j = 1, 2 \ldots, n$, the same as that of the weighted objective function of WOSA-**w**. For the SSAP mixed policy, we prove the expected rewards per task $R_w(\Phi 2)$ and $r_j(\Phi 2)$ for $j = 1, 2 \ldots, n$, have the same convergence rate as those under policy ($\Phi 1$). For the single-threshold mixed policy, although it requires little computational effort, the expected rewards per task $R_w(\Phi 3)$ and $r_j(\Phi 3)$ for $j = 1, 2 \ldots, n$, have a slower convergence rate than those under policy ($\Phi 1$), which is $O(\sqrt{\ln T}/\sqrt{T})$.

2.6.1 Convergence rates under the SSAP optimal policy

First, we provide the convergence rate of threshold values for the SSAP optimal policy defined by (2.6). The key technique is applying the properties of order statistics, since the threshold value $a_{\lceil T\theta \rceil,0}$ is the expected value of the $\lceil T\theta \rceil^{th}$ smallest combined value of T tasks, for some θ with $0 < \theta < 1$.

Lemma 2. *The threshold value $a_{\lceil T\theta \rceil,0}$ defined by (2.6) converges to G_{l+1} with an exponential rate as $T \to +\infty$, i.e.,*

$$|a_{\lceil T\theta \rceil,0} - G_{l+1}| \leq 2G_L \exp(-2T\Delta_\theta^2),$$

where

$$\Delta_\theta \triangleq \min\{F_{\mathcal{G}}(G_{l+1}) - \theta, \theta - F_{\mathcal{G}}(G_l)\}, \tag{2.17}$$

for $F_{\mathcal{G}}(G_l) < \theta < F_{\mathcal{G}}(G_{l+1})$ and $l = 0, 1, \ldots, L-1$.

Proof: See Appendix A.

Theorem 8 provides the convergence rate of the expected weighted reward per task $R_w(\Phi 1)$ as $T \to +\infty$.

Theorem 8. *The expected weighted reward per task* $R_w(\Phi 1)$ (2.2) *of WOSA-***w** *converges to* $\rho_w^\zeta(\mathbf{w})$ *with rate* $O(1/\sqrt{T})$ *as* $T \to +\infty$*, i.e.,*

$$|R_w(\Phi 1) - \rho_w^\zeta(\mathbf{w})| = O(\frac{1}{\sqrt{T}}).$$

Proof: See Appendix A.

Next, we derive convergence rates for the expected rewards per task of the n objective functions of MOSSAP. Lemma 3 provides the convergence rate of $b_{i,0}^j$, the expected value of the j^{th} component of the task value vector that results in the i^{th} smallest combined value. This rate will be used in the convergence rate analysis for $r_j(\Phi 1)$.

Lemma 3. *The value of* $b_{\lceil T\theta\rceil,0}^j$ *defined by (2.8) converges to* $\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{l+1}]$ *with an exponential rate as* $T \to +\infty$*, i.e.,*

$$|b_{\lceil T\theta\rceil,0}^j - \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{l+1}]| \leq 4A_M \exp(-2T\Delta_\theta^2),$$

where $A_M \triangleq \max_j A_j^{K_j}$ *and* Δ_θ *is given by* (2.17)*, for* $F_{\mathcal{G}}(G_l) < \theta < F_{\mathcal{G}}(G_{l+1})$*,* $l = 0, 1, \ldots, L-1$ *and* $j = 1, 2, \ldots, n$*.*

Proof: From the definition of $b_{\lceil T\theta\rceil,0}^j$ in (2.8),

$$\begin{aligned}
&|b_{\lceil T\theta\rceil,0}^j - \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{l+1}]| \\
\overset{(2.8)}{=}&|\mathbb{E}[\hat{\mathcal{A}}_T^{(j)(\lceil T\theta\rceil)}|\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}] - \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{l+1}]|\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}) \\
\leq&2A_j^{K_j}\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}) \\
\leq&4A_M \exp(-2T\Delta_\theta^2),
\end{aligned}$$

where $A_M = \max_j A_j^{K_j}$ and the last inequality follows from Lemma 2 with Δ_θ given by (2.17).

Theorem 9 provides the convergence rates of the expected rewards per task for each component of the reward vector of MOSSAP.

Theorem 9. *The expected reward per task* $r_j(\Phi 1)$ (2.1) *of MOSSAP converges to* $\rho_j^\zeta(\mathbf{w})$ *with rate* $O(1/\sqrt{T})$ *as* $T \to +\infty$*, i.e.,*

$$|r_j(\Phi 1) - \rho_j^\zeta(\mathbf{w})| = O(\frac{1}{\sqrt{T}}),\ \textit{for } j = 1, 2, \ldots, n.$$

Proof: See Appendix A.

2.6.2 Convergence rates under the SSAP mixed policy

Theorem 10 shows that the expected rewards per task under the SSAP mixed policy, $R_w(\Phi 2)$ and $r_j(\Phi 2)$ for $j = 1, 2, \ldots, n$, have the same convergence rates as those under the SSAP optimal policy, which are all $O(1/\sqrt{T})$.

Theorem 10. *The expected rewards per task $R_w(\Phi 2)$ (2.2) of WOSA-$\mathbf{w}$ and $r_j(\Phi 2)$ (2.1) of MOSSAP converge to $\rho_w^\zeta(\mathbf{w})$ and $\rho_j^\zeta(\mathbf{w})$ with rate $O(1/\sqrt{T})$ as $T \to +\infty$, respectively, i.e.,*

$$|R_w(\Phi 2) - \rho_w^\zeta(\mathbf{w})| = O(\frac{1}{\sqrt{T}}),$$
$$|r_j(\Phi 2) - \rho_j^\zeta(\mathbf{w})| = O(\frac{1}{\sqrt{T}}), \text{ for } j = 1, 2, \ldots, n.$$

Proof: See Appendix A.

2.6.3 Convergence rates under the single-threshold mixed policy

First, Lemma 4 provides a critical property under the single-threshold mixed policy, which is essential to the convergence rate analysis for policy ($\Phi 3$).

Lemma 4. *For a fixed T and $\eta = \lfloor T(1-\zeta) \rfloor$, define $U_s \triangleq \min\{k \in \mathbb{Z} : \sum_{t=1}^{k} X_t^{\Phi 3} = \eta\}$ and $U_{ns} \triangleq \min\{k \in \mathbb{Z} : \sum_{t=1}^{k}(1 - X_t^{\Phi 3}) = T - \eta\}$, with $X_t^{\Phi 3}$ given by ($\Phi 3$). Let $U_{min} = \min\{U_s, U_{ns}\}$. Then for any $\epsilon > 0$,*

$$\mathbb{P}\left(\frac{U_{min}}{T} < 1 - \epsilon\right) \to 0$$

with an exponential rate as $T \to +\infty$. Specifically, for $\epsilon > 0$ small,

$$\mathbb{P}\left(\frac{U_{min}}{T} < 1 - \epsilon\right) < (1 + e)\exp(-T\Delta_U^2),$$

where

$$\Delta_U \triangleq \sqrt{\min\{(\frac{\epsilon}{2})^2 \frac{1-\zeta}{\zeta}, \epsilon^2 \frac{\zeta}{1-\zeta}\}} = \epsilon\sqrt{\min\{\frac{1-\zeta}{4\zeta}, \frac{\zeta}{1-\zeta}\}}. \tag{2.18}$$

Proof: See Appendix A.

Theorem 11 provides the convergence rates of the expected rewards per task under the single-threshold mixed policy, $R_w(\Phi 3)$ and $r_j(\Phi 3)$ for $j = 1, 2, \ldots, n$, which are all $O(\sqrt{\ln T}/\sqrt{T})$.

Theorem 11. *The expected rewards per task* $R_w(\Phi 3)$ (2.2) *of WOSA-***w** *and* $r_j(\Phi 3)$ (2.1) *of MOSSAP converge to* $\rho_w^\zeta(\mathbf{w})$ *and* $\rho_j^\zeta(\mathbf{w})$ *with rate* $O(\sqrt{\ln T}/\sqrt{T})$ *as* $T \to +\infty$*, respectively, i.e.,*

$$|R_w(\Phi 3) - \rho_w^\zeta(\mathbf{w})| = O(\frac{\sqrt{\ln T}}{\sqrt{T}}),$$
$$|r_j(\Phi 3) - \rho_j^\zeta(\mathbf{w})| = O(\frac{\sqrt{\ln T}}{\sqrt{T}}),\ \textit{for } j = 1, 2, \ldots, n.$$

Proof: See Appendix A.

2.7 Trade-off Analysis and Generalization

In this section, we discuss the asymptotic results obtained in Sections 2.3, 2.4, 2.5 and 2.6 and their generalizations. First, we show the asymptotic expected reward per task for each component of the reward vector of Pareto optimal policies for MOSSAP captures the trade-off between the n objective functions defined by (2.1). Then we generalize these asymptotic results to MOSSAP with heterogeneous workers.

2.7.1 Trade-off analysis using asymptotic results of MOSSAP

Since the SSAP optimal policy, the SSAP mixed policy and the single-threshold mixed policy for WOSA-**w** achieve the same asymptotic expected reward per task for each component of the reward vector, we only discuss the trade-off under the SSAP optimal policies for different weight vector $\mathbf{w} \geq 0$. Note that each component of the task value vector is discrete with finite support, and hence the number of different values for $\rho_j^\zeta(\mathbf{w})$ under all optimal policies for WOSA with all $\mathbf{w} \geq 0$ is finite for $j = 1, 2, \ldots, n$ (see the expression given by (2.13)). This implies enumerating the asymptotic expected rewards per task under all such policies is feasible. [65] proves that when w_j is fixed for all $j = 1, 2, \ldots, k-1, k+1, \ldots, n$, $\rho_k^\zeta(\mathbf{w})$ increases when

w_k increases. Therefore, the range for obtainable asymptotic expected rewards per task can be obtained by choosing such weight vectors that $w_{j'}$ is set to be very large (zero) and $\{w_j\}_{j\neq j'}$ is set to be zero (very large), for $j' = 1, 2, \ldots, n$.

Recall that from the definition of $R_w(\Phi)$ (2.2), if $\mathbf{w}$ has only one non-zero component $w_j \neq 0$, denoted by $\mathbf{w}_{(j)}$, then $\max_{\Phi\in\Psi^\eta} R_w(\Phi) = \max_{\Phi\in\Psi^\eta} r_j(\Phi)$ and the resulting $\rho_w^\zeta(\mathbf{w}_{(j)})$ given by (2.11) is the optimal asymptotic expected reward per task for $r_j(\Phi)$. Define the *achievement ratio* for $r_j(\Phi)$ under an SSAP optimal policy for WOSA-$\mathbf{w}$ (denoted by Φ_w) as

$$\delta_j^\zeta(\mathbf{w}) \triangleq \lim_{T\to+\infty} \frac{r_j(\Phi_w)}{\max_{\Phi\in\Psi^{\lfloor T(1-\zeta)\rfloor}} r_j(\Phi)} = \frac{\lim_{T\to+\infty} r_j(\Phi_w)}{\lim_{T\to+\infty} \max_{\Phi\in\Psi^{\lfloor T(1-\zeta)\rfloor}} r_j(\Phi)} = \frac{\rho_j^\zeta(\mathbf{w})}{\rho_w^\zeta(\mathbf{w}_{(j)})}, \tag{2.19}$$

for $j = 1, 2, \ldots, n$, where $\rho_w^\zeta(\mathbf{w})$ and $\rho_j^\zeta(\mathbf{w})$ are given by (2.11) and (2.14), respectively. Therefore, for policy Φ_w, $\delta_j^\zeta(\mathbf{w})$ is the ratio of the asymptotic expected reward for $r_j(\Phi_w)$ under this policy to the optimal asymptotic expected reward for $r_j(\Phi)$. Clearly, $\delta_j^\zeta(\mathbf{w})$ is a function of the weight vector $\mathbf{w}$, with $0 \leq \delta_j^\zeta(\mathbf{w}) \leq 1$. In general, the magnitude of $\delta_j^\zeta(\mathbf{w})$ measures the closeness of $r_j(\Phi_w)$ under an SSAP optimal policy Φ_w for WOSA-$\mathbf{w}$ to the optima (the larger the closer). Therefore, the vector of $(\delta_1^\zeta(\mathbf{w}), \ldots, \delta_n^\zeta(\mathbf{w}))$ shows the trade-off between the n objective functions of MOSSAP under the SSAP optimal policies with different weight vectors.

2.7.2 Generalization of asymptotic results of MOSSAP

First we show that MOSSAP defined in Section 2.1 can be generalized to MOSSAP with multiple classes of workers (i.e., heterogeneous workers). Suppose there are $M \in \mathbb{Z}^+$ classes of workers. Each worker in class m has a success rate of τ^m, for $m = 1, 2, \ldots, M$. We assume $\tau^m > \tau^{m+1}$ for $m = 1, 2, \ldots, M-1$. Let N_m denote the number of workers in class m, and hence $\sum_{m=1}^M N_m = T$. Let $\zeta_m = \sum_{j=m+1}^M N_j/T$ denote the fraction of workers with a success rate strictly less than τ^m for $m = 1, 2, \ldots, M$ (set $\zeta_M = 0$). Then the objective function (2.1) becomes

$$r_j(\Phi) \triangleq \frac{1}{T}\mathbb{E}[\sum_{t=1}^{T}\sum_{m=1}^{M} X_{t,m}^\Phi \tau^m \mathcal{A}_j(t)], \quad \text{for } j = 1, 2, \ldots, n, \tag{2.20}$$

where $X^{\Phi}_{t,m}$ is the binary assignment variable for the t^{th} task under policy Φ: if $X^{\Phi}_{t,m} = 1$, then the t^{th} task is assigned to a worker in class m. The feasible region Ψ^{η}_{M} becomes

$$\Psi^{\eta}_{M} \triangleq \{\Phi : \sum_{t=1}^{T} X^{\Phi}_{t,m} = N_m, \text{ for all } m = 1, 2, \ldots, M, \text{ and } \sum_{m=1}^{T} X^{\Phi}_{t,m} \leq 1, \text{ for all } t\}.$$

Similarly, the objective function of WOSA-**w** becomes

$$R_w(\Phi) = \sum_{j=1}^{n} w_j r_j(\Phi) = \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\sum_{m=1}^{M} X^{\Phi}_{t,m}\tau^m \left(\sum_{j=1}^{n} w_j \mathcal{A}_j(t)\right)\right], \qquad (2.21)$$

for $\mathbf{w} \geq 0$.

The generalized results given in this section can be proven by following the proof of the corresponding theorem for MOSSAP with homogeneous workers, and hence we omit the proofs. The reason for these generalizations is that MOSSAP with homogeneous workers has essentially two classes of workers: original workers with a success rate of one and virtual workers with a success rate of zero. It has been proven that the difference between success rates of two classes of workers will not influence the optimal policy [4]. Therefore, for M classes of workers, the assignment for each task can be viewed as a $(M-1)$-stage assignment problem: at the m-th stage, the policy decides whether to assign the job to an available worker in class m or pass the job to the next stage (i.e., assigned to a worker with a lower success rate), for $m = 1, 2, \ldots, M-1$.

Pareto optimal policies for MOSSAP with multiple classes of workers can be generated using the weighted sum method by solving a sequence of WOSA-**w** problems, with the objective function given by (2.21). The optimal policy for WOSA-**w** is still a threshold-based policy with threshold values given by (2.6). However, $M-1$ threshold values are needed since there are M different worker classes. Specifically, this policy $\Phi 1_M$ is given by

$$X^{\Phi 1_M}_{t,m} = \begin{cases} 1, & \text{if } a_{T-t-\eta_m(t)+1,t} < \mathcal{G}(t) \leq a_{T-t-\eta_{m-1}(t)+1,t}, \\ 0 & \text{otherwise,} \end{cases} \qquad (\Phi 1_M)$$

$$\eta_m(t+1) = \eta_m(t) - X^{\Phi 1_M}_{t,m}, \qquad m = 1, 2, \ldots, M \text{ and } t = 1, 2, \ldots, T,$$

where $\eta_m(t)$ denote the number of available workers in class one to class m when the t^{th} task arrives, with $\eta_0(t) = 0$ for all t. Then Theorem 1 can be generalized to show that policy $\Phi 1_M$ is optimal for WOSA-**w** with multiple classes of workers (2.21) and $\{a_{i,0}\}_{i=1}^T$ are the expected combined values for the T tasks. Similarly, Corollary 1 can be generalized as

$$R_w(\Phi 1_M) = \max_{\Phi \in \Psi_M^\eta} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{T} \sum_{m=1}^{M} X_{t,m}^{\Phi} \tau^m \mathcal{G}(t)] = \frac{1}{T} \sum_{m=1}^{M} \tau_m \sum_{i=T-\sum_{j=1}^{m} N_j+1}^{T-\sum_{j=1}^{m-1} N_j} a_{i,0},$$

and

$$r_j(\Phi 1_M) = \frac{1}{T} \mathbb{E}[\sum_{t=1}^{T} \sum_{m=1}^{M} X_{t,m}^{\Phi 1_M} \tau^m \mathcal{A}_j(t)] = \frac{1}{T} \sum_{m=1}^{M} \tau_m \sum_{i=T-\sum_{j=1}^{m} N_j+1}^{T-\sum_{j=1}^{m-1} N_j} b_{i,0}^j,$$

for $j = 1, 2, \ldots, n$, where $\{a_{i,t}\}$ are defined by (2.6) and $\{b_{i,t}^j\}$ are defined by (2.8). We define that the summation $\sum_{i=i_{min}}^{i_{max}} n_i \triangleq 0$ if $i_{min} > i_{max}$ for any $\{n_i\}$.

The asymptotic results for MOSSAP with homogeneous workers in Sections 2.3, 2.5 and 2.6 can also be generalized to MOSSAP with heterogeneous workers. We assume the fraction of workers in each class is fixed as T approaches infinity (i.e., ζ_m remains the same for all m). Therefore, the optimal asymptotic expected weighted reward per task for WOSA-**w** is given by generalizing Theorem 2 as

$$\rho_w^M(\mathbf{w}) = \sum_{m=1}^{M} \tau^m \Bigg(\Big(\sum_{k=l_m+2}^{l_{m-1}} G_k p_{\mathcal{G}}(G_k) \Big) + q_m G_{l_m+1} p_{\mathcal{G}}(G_{l_m+1}) + (1 - q_{m-1}) G_{l_{m-1}+1} p_{\mathcal{G}}(G_{l_{m-1}+1}) \Bigg),$$

where

$$q_m = \frac{F_{\mathcal{G}}(G_{l_m+1}) - \zeta_m}{p_{\mathcal{G}}(G_{l_m+1})}, \text{ for } m = 1, 2, \ldots, M, \tag{2.22}$$

and $F_{\mathcal{G}}(G_{l_m}) \leq \zeta_m < F_{\mathcal{G}}(G_{l_m+1})$ for some $l_m \in \{0, 1, \ldots, L-1\}$ with $q_0 = 1$ and $l_0 = L$. The asymptotic expected reward per task for each component

of the reward vector of MOSSAP is hence

$$\rho_j^M(\mathbf{w}) = \sum_{m=1}^{M} \tau^m \Big(\mathbb{E}[\mathcal{A}_j(t) | G_{l_m+1} < \mathcal{G}(t) < G_{l_{m-1}+1}] \mathbb{P}_{\mathcal{G}}(G_{l_m+1} < \mathcal{G}(t) < G_{l_{m-1}+1}) + q_m \mathbb{E}[\mathcal{A}_j(t) | \mathcal{G}(t) = G_{l_m+1}] p_{\mathcal{G}}(G_{l_m+1}) + (1 - q_{m-1}) \mathbb{E}[\mathcal{A}_j(t) | \mathcal{G}(t) = G_{l_{m-1}+1}] p_{\mathcal{G}}(G_{l_{m-1}+1}) \Big),$$

which is obtained by generalizing Theorem 3. Corollary 2 and Lemma 1 can be generalized to MOSSAP with heterogeneous workers without any change.

There is no direct or simple way to generalize the SSAP mixed policy to MOSSAP with heterogeneous workers. However, generalizing the single-threshold mixed policy is direct and simple. The only change is that we need $M-1$ threshold values rather than a single one as in the case of MOSSAP with homogeneous workers. This policy, $\Phi 3_M$, is given as follows, which achieves the optimal asymptotic expected reward per task for WOSA-$\mathbf{w}$ and the same asymptotic expected reward per task for MOSSAP as policy $\Phi 1_M$.

$$X_{t,m}^{\Phi 3_M} = \begin{cases} 1, & \text{if } G_{l_m+1} < \mathcal{G}(t) < G_{l_{m-1}+1},\ \eta_m(t) > 0, \\ 1, & \text{with probability } q_m \text{ if } \mathcal{G}(t) = G_{l_m+1},\ \eta_m(t) > 0, \\ 1, & \text{with probability } (1 - q_{m-1}) \text{ if } \mathcal{G}(t) = G_{l_{m-1}+1},\ \eta_m(t) > 0, \\ 0, & \text{otherwise,} \end{cases} \tag{$\Phi 3_M$}$$

$$\eta_m(t+1) = \eta_m(t) - X_{t,m}^{\Phi 3_M}, \tag{2.23}$$

for $t = 1, 2, \ldots, T$ and $m = 1, 2, \ldots, M$ with $\{q_m\}$ given by (2.22).

Convergence rates for policies $\Phi 1$ and $\Phi 3$ can be directly generalized to policies $\Phi 1_M$ and $\Phi 3_M$, respectively, without any change.

CHAPTER 3

SCHEDULING C-BENEVOLENT JOBS ON UNWEIGHTED MACHINES

This chapter considers scheduling a sequence of C-benevolent jobs on multiple homogeneous machines. For two machines, we propose a 2-competitive *Cooperative Greedy* algorithm and provide a lower bound of 2 for the competitive ratio of any deterministic online scheduling algorithms on two machines. For multiple machines, we propose a *Pairing-m* algorithm, which is deterministic 2-competitive for even number of machines and randomized (2+2/m)-competitive for odd number of machines. We provide a lower bound of 1.436 for the competitive ratio of any deterministic online scheduling algorithms on three machines, which is the best known lower bound for competitive ratios of deterministic scheduling algorithms on three machines.

3.1 Formulation

This section provides variable definitions and clarifies notations for the online interval scheduling problem considered in the online interval scheduling problem for both cases of unweighted and weighted machines.

Consider a fixed set of machines. Let m denote the total number of machines. Each machine M_i has a positive weight, denoted by w_i, for $i = 1, 2, \ldots, m$. For the case of unweighted machines, set $w_i = 1$ for $i = 1, 2, \ldots, m$, without loss of generality. An *instance* is a sequence of N (not known *a priori*) arriving jobs, one after another, to be scheduled on one of the available machines. Let $\mathbf{I} = \{J_1, J_2, \ldots, J_N\}$ denote the list of arriving jobs, where J_j is a vector (defined in the following) of the j^{th} arriving job. One machine can execute at most one job at a time and one job can be assigned to at most one machine. The scheduling assignment is preemptive, and hence a scheduling assignment may be terminated before completion in favor of a later arriving job and is a *temporary* assignment. The terminated

job is said to be *aborted*.

A *job vector* $J_j = (a_j, l_j, v_j)$ is revealed upon the j^{th} job arrival, for $j = 1, 2, \ldots, N$. For each job vector, a_j denotes the *arrival time* of the j^{th} job, l_j denotes the *length* of the j^{th} job, and v_j denotes the *value* of the j^{th} job. Therefore, if a job is assigned to a machine, the *completion time* is defined as $f_j \triangleq a_j + l_j$. Moreover, we refer to the interval $[a_i, f_i)$ as the *job interval*. We assume that no two jobs share the same arrival time. If two jobs J_{j_1} and J_{j_2} satisfy $[a_{j_1}, f_{j_1}) \bigcap [a_{j_2}, f_{j_2}) \neq \emptyset$, then jobs J_{j_1} and J_{j_2} are said to *conflict* with each other.

The objective of this online scheduling problem is to maximize the total *reward* of completed jobs, subject to the constraint of the number of available machines. If a job J_j is assigned to the machine M_i, then the reward of this assignment is given by $r_{i,j} = v_j w_i$, which is gained only after completing the job. Therefore, the reward of any assignment terminated before execution is completed will be zero, and the job will be considered lost, since the termination of an assignment is irrevocable. We use this simple product form reward function in this chapter. Section 4.4 discusses the generalization of our results to other reward functions.

Let $OPT(\mathbf{I})$ denote the optimal reward for a job instance $\mathbf{I}$, which is obtained with the complete knowledge of $\mathbf{I}$ and hence is the optimal *off-line reward*. Let $R_{\mathcal{A}}(\mathbf{I})$ denote the reward obtained by algorithm $\mathcal{A}$ for a job instance $\mathbf{I}$. We employ the standard definition for competitive ratios, given in Definition 2.

Definition 2. *An online algorithm $\mathcal{A}$ is said to have a competitive ratio of γ if $R_{\mathcal{A}}(\mathbf{I}) \geq OPT(\mathbf{I})/\gamma$ for any job instance $\mathbf{I}$ generated by an adapted adversary.*

By Definition 2, $\gamma \geq 1$. It is already known that if the relationship between the length and the job value is arbitrary, no finite competitive ratio can be guaranteed [39]. Therefore, we focus on a special class of jobs, referred to as *C-benevolent jobs*, as introduced by [39]. For C-benevolent jobs, job values are a function of lengths (i.e., $v_i = g(l_i)$). Moreover, the function $g(l)$ is *C-benevolent* (see Definition 3). Note that this class of jobs fits the application for the aviation security screening problem. Although the exact relationship between the risk value and the screening time is indefinite, we have the following observations based on experience: (a) the risk value is an

increasing function of the screening time (i.e., a passenger with a higher risk value requires a longer screening time); (b) the risk value is a convex function of the screening time (i.e., screening a highly suspected passenger will gain a larger risk value than screening several low-risk passengers using the same amount of time). Therefore, the relationship between the risk value and the screening time satisfies C-benevolent conditions.

Definition 3. *A function $g(l)$ is said to be C-benevolent if $g(l)$ is a positive, convex, strictly increasing and continuous function of l. In other words, the convexity property of C-benevolent function $g(l)$ implies that*

$$g(a+\epsilon)+g(b-\epsilon) \leq g(a)+g(b),$$

for $0 < \epsilon \leq a \leq b$.

Note that C-benevolent jobs include jobs with values linearly proportional to lengths but do not include jobs with equal length and arbitrary values. However, our results can be applied to jobs with equal length and arbitrary values, as discussed in Section 4.4.

3.2 *Cooperative Greedy* Algorithm for Two Machines

This section considers two machines, which is the basic case for multiple machines. The analysis method used here provides insights for multiple machines, and this algorithm is later generalized to multiple machines in Section 3.4.

We propose a deterministic algorithm for two machines, referred to as the *Cooperative Greedy* algorithm. This algorithm is inspired by the 2-competitive algorithm given by [41], which is designed for jobs with values proportional to lengths (a special class of C-benevolent functions). Independent from our work, [45] uses the same idea in proposing a randomized 2-competitive algorithm for scheduling C-benevolent jobs on a single machine. The proof of Theorem 12 follows from the proof of *Theorem 3.3* [45] and hence is omitted here.

Theorem 12. *The Cooperative Greedy algorithm is 2-competitive for scheduling C-benevolent jobs on two machines.*

Algorithm 1 *Cooperative Greedy* Algorithm

Arbitrarily pick one machine as the primary machine (PM) and the other one as the secondary machine (SM).
for all job intervals J_i in an instance **I do**
  if PM has just completed executing some job **then**
    Switch the role of PM and SM.
    if PM is not executing any job **then**
      Assign J_i to PM, and let J_i be executed till it is completed.
    else
      Assign J_i to SM temporarily.
    end if
  else if PM is executing some job while SM is not executing any job **then**
    Assign J_i to SM temporarily.
  else if PM is executing some job and SM is also executing some job J_j **then**
    Abort and assign J_i to SM only if $v_i > v_j$.
  end if
end for

3.3 *Greedy-2* Algorithm for Multiple Machines

This section considers a *Greedy-2* algorithm for scheduling on multiple machines and proves that it is 4-competitive. An example is provided to show that the analysis for the *Greedy-2* algorithm is tight. We provide a different proof than the one in [46].

3.3.1 A 4-competitive approximation algorithm

This section provides an approximation algorithm for multiple machines, the Multiple Greedy-2 algorithm, described as follows. We prove that the Multiple Greedy-2 algorithm is 4-competitive for C-benevolent job sequences by extending the deterministic results of [39] to scheduling algorithms on multiple machines.

We first clarify some notations, which follow those from [39], as shown in Figure 3.1. Consider a job J completed under the *Greedy-2* algorithm on a single machine. Then, all jobs that are temporarily assigned by the *Greedy-2* algorithm but later aborted directly or indirectly in favor of J are

Algorithm 2 *Greedy-2* Algorithm

Let $S_k(t)$ denote the job being executed on machine M_k at time t, for $k = 1, 2, \ldots, m$.
for all job intervals J_i in an instance **I do**
  if $v_i \geq 2 \min_k v(S_k(a_i))$ **then**
    $k' = \arg\min_k v(S_k(a_i))$
    Abort $S_{k'}(t)$ and assign J_i to machine $M_{k'}$.
    Update $S_{k'}(t)$, i.e., $S_{k'}(t) = J_i$, for $t \in [a_i, a_i + l_i)$.
  else
    Discard J_i.
  end if
end for

called *predecessors* of J. The job not assigned by the *Greedy-2* algorithm that arrives during the execution of J and has the largest completion time is called the *successor* of J. The set of jobs consisting of all predecessors of J, J, and the successor of J is referred to as the *segment of* J. Let $G(J)$ denote the segment of J under the *Greedy-2* algorithm. Then, from *Observation 3.1* in [39], a job instance can be divided into non-overlapping segments of all completed jobs under the *Greedy-2* algorithm (i.e., no job arrives during the gap between subsequent segments, if such a gap exists). Next, we mark the arrival times of all predecessors of J, the arrival time and completion time of J, and the completion time of the successor of J. These time points are referred to as the *marked time points* of a segment. Number these time points starting from the very last time point of a segment to the beginning of the segment, namely backwards in time ($b_0, b_1, \ldots$ and $c_0, c_1, \ldots$ as shown in the Figure 3.1). A job K that arrives during the interval defined by $G(J)$, which starts from the arrival time of the very first processor and ends at the completion time of J, is called *belonging to* $G(J)$. Let $H(J)$ denote the set of jobs belonging to the segment of J.

Before analyzing the *Greedy-2* algorithm, we first give Proposition 2 for the *Greedy-2* algorithm on a single machine, which is an important property of Greedy algorithms for C-benevolent job sequences. Note that Proposition 2 is stronger than *Theorem 3.2* in [39] since it includes infeasible schedules, while *Theorem 3.2* in [39] only considers feasible schedules.

Proposition 2. *Let* $\{b_i\}_{i=0}^{n}$ *denote the marked time points of the segment of a completed job* J*. Then for any set of jobs* $I \subset H(J)$*, the sum of values*

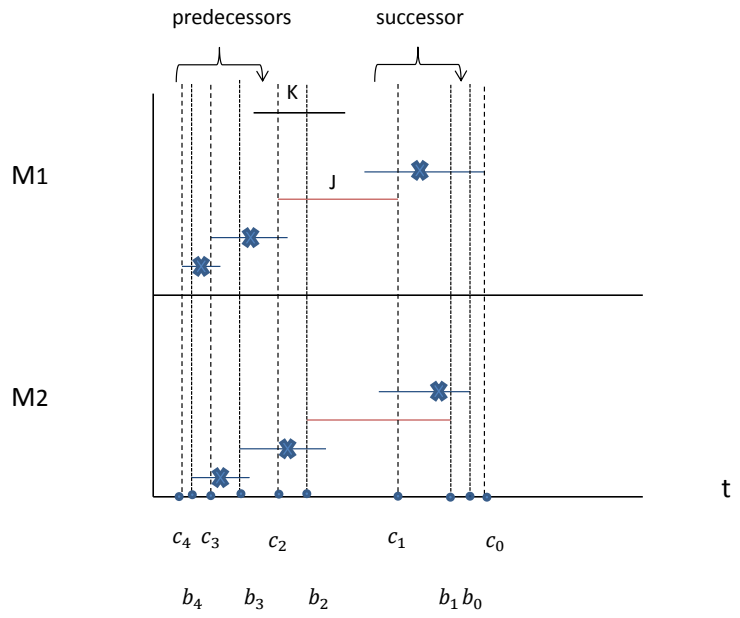

Figure 3.1: Segments of completed jobs under the *Greedy-2* algorithm.

of the jobs in I is no greater than $4v(J)$, as long as the following conditions are satisfied: (a) no overlapping interval exists within a job (i.e., for any job $J_{i_1} \in I$, there is no job $J_{i_2} \in I$ such that $[a_{i_1}, f_{i_1}) \subset [a_{i_2}, f_{i_2})$); (b) for any interval defined by two marked time points $[b_{i+1}, b_i)$, only one job that arrives during this interval and completes after b_i can be selected in I, for $i = 1, 2 \ldots, n-1$.

Proof: Since C-benevolent functions are increasing, we assume that jobs in I cover the entire interval of $[b_n, b_0]$ (this only increases the sum of values of the jobs in I, since C-benevolent functions are convex and increasing). First consider those jobs in I that cover a marked time point. Note that a job can cover at most one marked time point. Then, for any job $K \in I$ and $K \notin G(J)$, if K covers some marked time point b_i, then $v(K) \leq 2^{2-i}v(J)$, for $i = 1, 2, \ldots, n-1$. If $K \in G(J)$ such that K arrives at some marked time point $b_{i'}$, pick $b_{i'}$ as the marked time point that K covers, and we still have $v(K) \leq 2^{2-i'}v(J)$, for $i' = 1, 2, \ldots, n$. For those jobs in I that cover no marked time point, since overlapping is not permitted within any job in I (by condition (a)), then the jobs covering no marked point only exist in the gap between those jobs in I that cover some marked time point. Therefore, since C-benevolent functions are increasing, the sum of values of the jobs in

I is

$$\sum_{K \in I} v(K) \leq \sum_{i=1}^{n-1} 2^{2-i} v(J) \leq 4v(J).$$

■

Let $\mathbf{G}_k$ denote the set of jobs assigned to machine M_k by the *Greedy-2* algorithm, including both aborted jobs and completed jobs, for $k = 1, 2, \ldots, m$. Let OPT_k denote the set of jobs scheduled on M_k in the optimal schedule for m machines, for $k = 1, 2, \ldots, m$. The competitive ratio of the *Greedy-2* algorithm is given in Theorem 13.

Theorem 13. *The Greedy-2 algorithm is 4-competitive for C-benevolent job sequences on multiple machines.*

Proof: First, we process all the jobs in $\{OPT_k\}_{k=1}^{m}$ by *checking* and *grouping* as follows. For any job J in OPT_k, for $k = 1, 2, \ldots, m$, if $J \in \mathbf{G}_{k'}$ for some $k' = 1, 2, \ldots, m$, we say job J in $\mathbf{G}_{k'}$ is *checked for the whole interval of* J. Let G_C denote the set of all the checked jobs in $\bigcup_k \mathbf{G}_k$. Next, consider a job $J \in \bigcup_k OPT_k \setminus G_C$. Since $J \notin \bigcup_k \mathbf{G}_k$, all the machines must be executing some other job when J arrives. Let $K_k(J)$ denote the job being executed on machine M_k when J arrives, for $k = 1, 2, \ldots, m$. Therefore, $v(J) < 2v(K_k(J))$, for $k = 1, 2, \ldots, m$. We want to *group* J with one of the segments that $K_k(J)$ belongs to. The grouping option is different for the two cases: (a) there is some job $K_{\bar{k}}(J)$ that is completed before the completion time of J; (b) no $K_k(J)$ is completed before the completion time of J. For case (a), group J with the segment that $K_{\bar{k}}(J)$ belongs to. If there are more than one such $K_{\bar{k}}(J)$, choose one segment arbitrarily and group J with it. For case (b), then there must exist at least one machine $M_{\hat{k}}$ such that one of the following conditions holds: (a) the completion time of J is earlier than that of $K_{\hat{k}}(J)$, and $K_{\hat{k}}(J)$ is not checked for the interval of $[a(J), a(J)+l(J))$; (b) the completion time of J is later than that of $K_{\hat{k}}(J)$, and both $K_{\hat{k}}(J)$ and the subsequent job that aborts $K_{\hat{k}}(J)$ are not checked for the interval of $[a(J), a(J) + l(J))$. The existence of these conditions is due to the feasibility of J on one of the machines (i.e., $J \in \bigcup_k OPT_k$). If condition (a) holds, group J with the segment that $K_{\hat{k}}(J)$ belongs to and check $K_{\hat{k}}(J)$ for the interval of $[a(J), a(J) + l(J))$. If condition (b) holds, group J with the segment that $K_{\hat{k}}(J)$ belongs to and check $K_{\hat{k}}(J)$ and the subsequent job that aborts $K_{\hat{k}}(J)$ for the interval of $[a(J), a(J) + l(J))$ (if there is more than one machine that

fulfills condition (a) or (b), choose one arbitrarily). In this way, each job in $\bigcup_k OPT_k$ is grouped to one segment, either by checking or grouping.

Then, we consider each grouped segment S on any machine. According to our checking and grouping strategy, a job in the segment S is either checked for the whole interval or checked partially. Moreover, no partial interval within a job can be checked more than once, which satisfies condition (a) in Proposition 2. In addition, the grouped jobs in a segment satisfy condition (b) in Proposition 2. Therefore, from Proposition 2, the sum of values of the jobs in $\bigcup_k OPT_k$ grouped to each segment is no greater than four times the value of the job completed by the *Greedy-2* algorithm in that segment on the particular machine. That is, the 4-competitive ratio holds for any grouped segment. Since a job instance can be divided into non-overlapping segments, the *Greedy-2* algorithm is 4-competitive for C-benevolent job sequences.

■

3.3.2 Tightness of analysis for the *Greedy-2* algorithm

We provide a small example of a job instance that drives the competitive ratio of the *Greedy-2* algorithm arbitrarily close to 4 and hence shows that our analysis for the *Greedy-2* algorithm is tight.

Example 1. *Suppose a job instance is constituted of m identical pairs of geometric-sets of stage-N and main-sets of stage-$(N+1)$. A geometric-set of stage-N is a sequence of N jobs, where the arrival time of the subsequent job is immediately before the completion time of the previous job with a small overlapping, and values of subsequent jobs increase by a factor of two. More precisely, let $\{(a_i, l_i, v_i)\}_{i=1}^{N}$ be a geometric-set of stage-N. Then the following conditions are satisfied: (a) $v_{i+1} = 2v_i$, for $i = 1, 2, \ldots, N-1$; (b) $a_{i+1} = a_i + l_i - \epsilon$, for $i = 1, 2, \ldots, N-1$, where ϵ is chosen to be sufficiently small. Therefore, only the last job can be completed in a geometric-set under the Greedy-2 algorithm. A main-set of stage-$(N+1)$ is a sequence of $N+1$ jobs paired with the geometric-set, which is feasible on a machine but is neglected due to the existence of the corresponding geometric-set. For the geometric-set of stage-N $\{(a_i, l_i, v_i)\}_{i=1}^{N}$, the main-set may be constructed as $\{(a'_i, l'_i, v'_i)\}_{i=1}^{N+1}$ satisfying the following conditions: (a) $a'_i = a_i + \epsilon$, for $i = 1, 2, \ldots, N-1$, and $a_{N+1} = a_N + l_N - \epsilon$; (b)$a'_i + l'_i = a_i + l_i$, for*

$i = 1, 2, \ldots, N-1$, and $a'_N + l'_N = a_N + l_N - \epsilon$; (c) $v'_i = v_i - \delta$, for $i = 1, 2, \ldots, N$, and $v_{N+1} = 2v_N - \delta$. Note that δ is determined by ϵ through the C-benevolent functions, and ϵ can be selected to be sufficiently small such that δ is sufficiently small. We set subsequent pairs of the m pairs of geometric-sets and main-sets to have a slight difference τ in arrival times such that $m\tau < \epsilon$, and hence τ is negligible. Therefore, under the Greedy-2 algorithm, each machine will only select intervals from the geometric-sets and neglect jobs in the main-sets. Moreover, only the last job in each geometric-set is completed, and hence the reward for the Greedy-2 algorithm is mv_N. However, the optimal schedule should be scheduling all the jobs in the m main-sets, and hence the optimal reward is $m\sum_{i=1}^{N+1} v'_i = m\left(\sum_{i=1}^{N} v_i + 2v_N - (N+1)\delta\right) = m\left((4 - 2^{-(N-1)})v_N - (N+1)\delta\right)$. Therefore, by setting N large and $(N+1)\delta \ll v_N$, the competitive ratio of the Greedy-2 algorithm can be made arbitrarily close to 4.

3.4 *Pairing-m* Algorithm for Multiple Machines

This section considers scheduling C-benevolent jobs on multiple machines (i.e., $m \geq 3$). We generalize the *Cooperative Greedy* algorithm for two machines to even and odd number of machines, respectively. The algorithm is referred to as the *Pairing-m* algorithm. We show that the *Pairing-m* algorithm is 2-competitive for even number of machines and (2+2/m)-competitive for odd number of machines. For even number of machines, the *Pairing-m* algorithm is deterministic. For odd number of machines, the *Pairing-m* algorithm is randomized.

3.4.1 *Pairing-m* algorithm for even number of machines

Let $m = 2k$, where $k \in \mathbb{Z}^+$. The *Pairing-m* algorithm works similarly to the *Cooperative Greedy* algorithm. It dynamically pairs up m machines, $2k$ as the primary machines (PM) and the other as the secondary machines (SM). The pairing between machines is not fixed and changed at the completion of jobs assigned on PMs. First, we divide the time axis into *sections*: the time interval starts from one of PMs gets assigned a job till the first time all machines are available. Therefore, the time axis can be divided into non-

overlapping sections with no job arriving between two successive sections. We describe how the *Pairing-m* algorithm assigns jobs in each section in the following.

Consider any section. In the beginning, we pick k machines as PMs and the other k machines as SMs arbitrarily. Suppose M_1 to M_k are PMs and M_{k+1} to M_{2k} are SMs. The *Pairing-m* algorithm starts by assigning arriving jobs to an available PM until all PMs are busy. If this never happens, then the *Pairing-m* algorithm completes all arrived jobs in this section and hence is the same as the optimal schedule. Otherwise, let $J_1, J_2, \ldots$ denote the jobs completed by the *Pairing-m* algorithm, indexed in the increasing order of completion times.

We further divide the section into non-overlapping *segments* based on the completion times of jobs assigned to PMs: the first segment is defined as the time interval between the start of the section and the completion time of the J_1; successive segments are defined as the time interval between two subsequent completion times of jobs assigned on PMs. Therefore, for instance, $[f(J_i), f(J_{i+1}))$ is a segment for $i = 1, 2, \ldots$. In each segment, jobs assigned on the k PMs will be guaranteed completion (remain un-preempted). If there is at least one available PM in the segment, then assign jobs arriving in this segment to these available PMs until all PMs are busy. If there is no available PM in the segment, then the k SMs greedily schedule jobs arriving in this segment: a newly arrived job J'_j is only scheduled on some SM if $v(J'_j) > \min v(J_{SM})$, where $\min v(J_{SM})$ is the minimum value of jobs executed on SMs when J'_j arrives (if a machine has no job assigned to it, we consider it as executing a virtual job of value zero and length zero). In this way, the k SMs will be assigned jobs the with the top k values out of all the jobs arriving in the segment (if there are $k' < k$ jobs arriving in the segment, we consider another $k - k'$ virtual jobs with value zero and lengh zero arriving in this segment). At the beginning of each segment, two machines (one PM and one SM) switch their roles: the PM which just completes its assigned job becomes an available SM, and the SM assigned the largest-value job among all jobs currently executed on SMs becomes an PM. Therefore, the *Pairing-m* algorithm keeps k PMs and k SMs in each time segment. This process continues till all $2k$ machines are available again, which is the end of a section. When the next job arrives, a new section begins in the same way.

Theorem 14. *The Pairing-m algorithm is 2-competitive for scheduling C-benevolent jobs on even number of machines.*

Proof: Since a job sequence can be divided into non-overlapping sections, we compute the competitive ratio of the Pairing-m algorithm in any section, which is the same as the competitive ratio of the Pairing-m algorithm for the job sequence.

Let $OPT(M_i, n)$ denote the optimal schedule on machine M_i (with abuse of notation, $OPT(M_i, n)$ also denotes the value for the optimal schedule depending on the context) when the Pairing-m algorithm completes n jobs in a section, for $i = 1, 2, \ldots, m$ and $n \geq 1$. Let $\{OPT_i(n)\}$ denote the order statistics of $\{OPT(M_i, n)\}$ such that $OPT_i(n) \leq OPT_{i-1}(n)$ for $i = 2, 3, \ldots, m$. Let $\{J_1, J_2, \ldots, J_n\}$ denote jobs completed by the Pairing-m algorithm, indexed in the increasing order of completion times with $f(J_1) \leq f(J_2) \leq \ldots \leq f(J_n)$. Note that the index of these jobs may not coincide with their arrival orders. Therefore, the completion time of the last job in the optimal schedule $OPT(M_i, n)$ is less than $f(J_n)$ for $i = 1, 2, \ldots, n$.

We prove $\sum_{i=1}^{k} OPT_i(n) \leq \sum_{j=1}^{n} v(J_j)$ by induction on the total number of jobs completed by the Pairing-m algorithm in a section. Consider the base case of $n \leq k$. Then, there are only k jobs arriving in this section, and all jobs are completed by PMs. Therefore, $\sum_{i=1}^{k} OPT_i(n) \leq \sum_{j=1}^{n} v(J_j)$ holds trivially.

Assume $\sum_{i=1}^{k} OPT_i(n) \leq \sum_{j=1}^{n} v(J_j)$ holds for some n. We want to show that $\sum_{i=1}^{k} OPT_i(n+1) \leq \sum_{j=1}^{n+1} v(J_j)$ holds. Consider the last completed job J_{n+1} by the Pairing-m algorithm. We compare the set of jobs arriving in this section $\mathbf{I}_{n+1}$ with $\{J_1, J_2, \ldots, J_{n+1}\}$ completed by the Pairing-m algorithm and another set of jobs arriving in this section $\mathbf{I}_n$ constructed as follows, with $\mathbf{I}_n \subset \mathbf{I}_{n+1}$. Let S_{n+1} denote the segment where J_{n+1} arrives and $\mathbf{I}_s$ denote the set of jobs that arrive in segment S_{n+1}. Note that no jobs arrive after segment S_{n+1}. Then define $\mathbf{I}_n \triangleq \mathbf{I}_{n+1} \setminus \mathbf{I}_s$. The Pairing-m algorithm will complete at most n jobs for the set of jobs $\mathbf{I}_n$ in this section. Let $\{J'_1, J'_2, \ldots, J'_n\}$ denote the set of jobs (indexed in increasing order of completion times) completed by the Pairing-m algorithm for $\mathbf{I}_n$ (if the number of completed jobs is smaller than n, append virtual jobs with value zero and length zero to the end).

If $\mathbf{I}_s = \{J_{n+1}\}$, then $J_i = J'_i$ for $i = 1, 2, \ldots, n$. By the induction assump-

tion,

$$\sum_{i=1}^{k} OPT_i(n+1) \leq \sum_{i=1}^{k} OPT_i(n) + v(J_{n+1}) \leq \sum_{j=1}^{n+1} v(J_j).$$

Otherwise, $\mathbf{I}_s$ contains more than one job. Let $\{K'_1, K'_2, \ldots, K'_k\} \subset \{J'_1, J'_2, \ldots, J'_n\}$ denote the k jobs assigned to the k SMs at the end of segment S_{n+1} for $\mathbf{I}_n$ (if some SM is available during this segment, then we consider this SM as executing a virtual job with value zero and length zero). Then $\{K'_1, K'_2, \ldots, K'_k\}$ will be completed by the Pairing-m algorithm for $\mathbf{I}_n$ since no more jobs arrive. However, for $\mathbf{I}_{n+1}$, these k SMs will be updated with jobs with the top k values out of $\mathbf{I}_s \bigcup \{K'_1, K'_2, \ldots, K'_k\}$. Let $\{K_1, K_2, \ldots, K_k\}$ denote the k jobs being executed on the k SMs at the end of segment S_{n+1} for $\mathbf{I}_{n+1}$. Then $J_{n+1} \in \{K_1, K_2, \ldots, K_k\}$. Consider $OPT_i(n+1)$, for $i = 1, 2, \ldots, k$. If there exists K_j such that $K_j \in OPT_i(n+1)$, then define $OPT'_i(n+1) \triangleq OPT_i(n+1) \setminus K_j$. Otherwise, if $OPT_i(n+1)$ does not contain any K_j, then if $OPT_i(n+1)$ does not schedule any job that conflicts with any K_j, then $OPT'_i(n+1) \triangleq OPT_i(n+1)$. Otherwise, suppose $OPT_i(n+1)$ schedules a set of jobs $\{H_j^i\}_{i=1}^{l_j}$ (indexed by i in the increasing order of completion times), which conflict with some K_j. Then, the completion time of $H_j^{l_j-1}$ is within the segment S_{n+1}, since $H_j^{l_j}$ arrives in the segment of S_{n+1}. By the scheduling policy of the Pairing-m algorithm, $v(H_j^i) < v(K_j)$ for any i. Define $OPT'_i(n+1) \triangleq OPT_i(n+1) \setminus H_j^{l_j}$. Consider the set of schedules $\{OPT'_i(n+1)\}_{i=1}^{k}$. Then the largest completion time of jobs in schedules $\{OPT'_i(n+1)\}_{i=1}^{k}$ is within S_{n+1}. Therefore, from the induction assumption,

$$\sum_{i=1}^{k} OPT'_i(n+1) \leq \sum_{j=1}^{n} v(J'_j) - \sum_{j=1}^{k} v(K'_j),$$

which follows from the fact that the largest completion time of jobs in schedules $\{OPT'_i(n+1)\}_{i=1}^{k}$ is already covered by the job completed at the end of segment S_{n+1}, and the completion times of $\{K'_1, K'_2, \ldots, K'_k\}$ are all beyond

S_{n+1}. Therefore,

$$\sum_{i=1}^{k} OPT_i(n+1) \leq \sum_{i=1}^{k} OPT'_i(n+1) + \sum_{j=1}^{k} v(K_j)$$
$$\leq \sum_{j=1}^{n} v(J'_j) - \sum_{j=1}^{k} v(K'_j) + \sum_{j=1}^{k} v(K_j) \leq \sum_{j=1}^{n+1} v(J_j),$$

where: the first inequality follows from the construction of $\{OPT'_i(n+1)\}_{i=1}^{k}$; the last inequality follows from $\sum_{j=1}^{k} v(K'_j) \leq \sum_{j=1}^{k} v(K_j)$ and $\{J_j\}_{j=1}^{n} \setminus \{K_i\}_{i=1}^{k} = \{J'_j\}_{j=1}^{n} \setminus \{K'_i\}_{i=1}^{k}$ from the construction of $\mathbf{I}_n$. ■

3.4.2 *Pairing-m* algorithm for odd number of machines

When the number of machines is odd, the *Pairing-m* algorithm for even number of machines cannot be directly applied. To overcome this difficulty, we introduce randomization and generalize the *Pairing-m* algorithm for even number of machines to odd number of machines.

Let $m = 2k+1$ for $k \in \mathbb{Z}^+$. We create a virtual machine, add it to the pool of real machines and treat this virtual machine the same as real machines. Then the *Pairing-m* algorithm can be applied to these $m+1$ machines. Let $OPT(M_i, \mathbf{I})$ and $P(M_i, \mathbf{I})$ denote the optimal schedule and the schedule using the Pairing-(m+1) algorithm on machine M_i for instance $\mathbf{I}$, respectively, for $i = 1, 2, \ldots, m+1$ (machine M_{m+1} is the virtual machine). Then arbitrarily pick m schedules out of $\{P(M_i, \mathbf{I})\}_{i=1}^{m+1}$ with equal probability, and schedule jobs to the m real machines according to these m selected schedules. This algorithm is referred to as the *Pairing-m* algorithm for odd-number of machines.

Theorem 15. *The Pairing-m algorithm is* $2+2/m$*-competitive for scheduling C-benevolent jobs on odd number of machines.*

Proof: Let $\{OPT_i(\mathbf{I})\}$ denote the order statistics of $\{OPT(M_i, \mathbf{I})\}$, with $OPT_i(\mathbf{I}) \geq OPT_{i+1}(\mathbf{I})$ for $i = 1, 2, \ldots, m$ (with abuse of notation, $\{OPT(M_i, \mathbf{I})\}$ also denotes the value of the schedule depending on the con-

text). Then from Theorem 14,

$$\sum_{i=1}^{k+1} OPT_i(\mathbf{I}) \leq \sum_{i=1}^{m+1} v(P(M_i, \mathbf{I})),$$

where $v(P(M_i, \mathbf{I}))$ denotes the total value of completed jobs by schedule $P(M_i, \mathbf{I})$. Since the Pairing-m algorithm randomly selects m schedules from $\{P(M_i, \mathbf{I})\}_{i=1}^{m+1}$ with equal probability, then the expected reward using the Pairing-m algorithm, denoted by $R_m(\mathbf{I})$, is lower bounded by

$$R_m(\mathbf{I}) = \frac{m}{m+1} \sum_{i=1}^{m+1} v(P(M_i, \mathbf{I})) \geq \frac{m}{m+1} \sum_{i=1}^{k+1} OPT_i(\mathbf{I}).$$

The optimal reward on m machines for instance $\mathbf{I}$, denoted by $O_m(\mathbf{I})$, is upper bounded by

$$O_m(\mathbf{I}) \leq \sum_{i=1}^{m+1} OPT_i(\mathbf{I}) \leq 2 \sum_{i=1}^{k+1} OPT_i(\mathbf{I}).$$

Therefore, the competitive ratio for the Pairing-m algorithm on odd number machines is given by $2(m+1)/m = 2 + 2/m$. ■

3.5 Lower Bounds for Competitive Ratios

This section gives lower bounds of 2 and 1.436 for the competitive ratio of any deterministic algorithm for scheduling C-benevolent jobs on two and three machines, respectively. Since *Cooperative Greedy* algorithm is 2-competitive for two machines, it is the best obtainable deterministic algorithm for scheduling C-benevolent jobs on two machines. From Theorem 15, the competitive ratio of the *Pairing-m* algorithm on three machines is $2 + 2/3 = 2.67$. Although there is still a gap between the competitive ratio of our proposed algorithm and the lower bound for three machines, the *Pairing-m* algorithm is the first-known 2.67-competitive randomized algorithm for scheduling C-benevolent jobs on three machines.

Theorem 16 gives a lower bound for any deterministic algorithms on two machines. The proof of Theorem 16 uses the same technique as the proof of *Theorem 6* [40], and hence, is omitted here.

Theorem 16. *No deterministic algorithm for C-benevolent jobs on two machines can achieve a competitive ratio lower than 2.*

Theorem 17 provides a lower bound for any deterministic algorithm for scheduling C-benevolent job sequences on three machines. We use an approach similar to that in [40], but we handle more complicated cases for three machines.

We use the *W-set*, the job set originally defined in [39], to prove this upper bound. A W-set of jobs is defined as a sequence of jobs that satisfy the following conditions: (a) jobs arrive in sequence but conflict with one another within the set; (b) the value of the first arriving job is set to be one; (c) the values of subsequent jobs differ from each other by a small amount $\delta > 0$ (monotonically increasing); (d) the arrival times of subsequent jobs differ from each other by a small time $\epsilon > 0$. Let $\bar{v}$ denote the value of the last job in a W-set (also the largest job value in this W-set by construction). Note that by setting the values of ϵ and δ, then $\bar{v}$ can be made to be arbitrarily large for C-benevolent job sequences.

Theorem 17. *No deterministic algorithm for C-benevolent job sequences on three machines can achieve a competitive ratio lower than 1.436.*

Proof: We prove this lower bound by considering a sequential game between a deterministic algorithm and an adaptive adversary. We will show that there exists a strategy for the adversary to drive the inverse of the competitive ratio of any deterministic algorithm to $0.696 + \zeta$, where $\zeta > 0$ can be arbitrarily small.

First, the adversary generates three identical W-sets, with sets arriving as one slightly after another by a delay of $\epsilon' \ll \epsilon$. We make this difference between these three sets only to conform to the assumption that no two jobs share the same arrival time; we will ignore this ϵ' difference in the arrival times of two jobs in the following. Now a deterministic algorithm, denoted by A, has several choices of which jobs to assign to each machine. Let $J_x(x)$, $J_y(y)$, and $J_z(z)$ denote the jobs (job values) that A assigns to M_1, M_2, and M_3, respectively. Let OPT denote the optimal reward and $r(A)$ denote the reward for algorithm A. Then the inverse of the competitive ratio of A is $\gamma_A = r(A)/OPT$. The adversary will react adaptively to different choices

made by A, as discussed case by case.

Note that if at least one machine does not have any job that has been scheduled by A, then γ_A should be no greater than $2/3 < 17/24$. Moreover, each machine can have at most one job from the three W-sets since jobs in these W-sets conflict with each other. Therefore, we assume that all three machines have one job from the three W-sets scheduled by A in the following.

Case 1 All of the scheduled jobs have a value of one. That is, $x = y = z = 1$. In this case, the adversary generates no more jobs. Therefore, $OPT = 3\bar{v}$, $r(A) = 3$, and hence, $\gamma_A = 1/\bar{v} \leq 1/2$, for $\bar{v} \geq 2$.

Case 2 Two of the scheduled jobs have a value of one. Suppose $x = y = 1$ and $z > 1$. In this case, the adversary generates no more jobs. Therefore, $OPT = 3\bar{v}$, $r(A) = 2+z$, and hence $\gamma_A = (2+z)/(3\bar{v}) \leq 1/3+2/(3\bar{v}) \leq 2/3$, for $\bar{v} \geq 2$.

Case 3 One of the scheduled jobs has a value of one. Suppose $x = 1$. Then we have two sub-cases for the other two scheduled jobs : (a) $y = z$ and (b) the value of one job is strictly smaller than the other. For sub-case (b), without loss of generality, suppose $y < z$. If $y \leq \bar{v}/2$ and $z \leq \bar{v}/2$, then $\gamma_A \leq 1/2$. Therefore, we consider $y = z > \bar{v}/2$ for case (a) and $z > \bar{v}/2$ for case (b).

For Case 3 (a), the adversary generates three additional identical jobs that arrive right before the completion time of J_y and J_z but after the completion time of the preceding job in the W-set (jobs that that arrive right before the completion time of some job J but after the completion time of the job preceding J in the W-set are referred to as the *challenger jobs* for J). The values of these three new jobs are all equal to y. Then, the reward for A is at most $r(A) = 1+3y$ (since job J_x does not conflict with the new arrivals). However, the optimal reward is $OPT = 3(2y - \delta)$. Therefore, $\gamma_A = (1 + 3y)/\left(3(2y - \delta)\right) \leq 2/3 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large.

For Case 3 (b), if $y > z/2$, then the adversary generates three additional challenger jobs with value z for J_y. Then, the reward for A is at most $r(A) = 1 + 3z$. However, the optimal reward is $OPT = 3(y + z - \delta)$. Therefore, $\gamma_A = (1 + 3z)/\left(3(y + z - \delta)\right) \leq 2/3 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large. Otherwise, if $y \leq z/2$, then the adversary

generates three additional challenger jobs with value z for J_z. Then, the reward for A is at most $r(A) = 1+y+3z$. However, the optimal reward is $OPT = 3(z + z - \delta)$. Therefore, $\gamma_A = (1 + y + 3z)/\left(3(z + z - \delta)\right) \leq 7/12 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large.

Case 4 None of the scheduled jobs has a value of one. That is, $x, y, z > 1$. In this case, we have four sub-cases: (a) All the scheduled jobs have the same value, $x = y = z$. (b) Two of the scheduled jobs have the same value, and this value is greater than the other job value; suppose $x < y = z$. (c) Two of the scheduled jobs have the same value, and this value is smaller than the other job value; suppose $x = y < z$. (d) None of the scheduled jobs has the same value; suppose $x < y < z$. We provide an upper bound for inverse of the competitive ratio γ_A for each sub-case separately.

For Case 4 (a), the adversary generates three additional challenger jobs with value x for J_x. Then, the reward for A is at most $r(A) = 3x$. However, the optimal reward is $OPT = 3(2x - \delta)$. Therefore, $\gamma_A = 3x/\left(3(2x - \delta)\right) \leq 1/2 + \zeta$, for δ sufficiently small.

For Case 4 (b), if $x > y/2$, then the adversary generates three additional challenger jobs with value y for J_x. Then, the reward for A is at most $r(A) = 3y$. However, the optimal reward is $OPT = 3(x + y - \delta)$. Therefore, $\gamma_A = 3y/\left(3(x + y - \delta)\right) \leq 2/3 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large. Otherwise, if $x \leq y/2$, then the adversary generates three additional challenger jobs with value y for J_y. Then, the reward for A is at most $r(A) = x+3y$. However, the optimal reward is $OPT = 3(2y - \delta)$. Therefore, $\gamma_A = (x + 3y)/\left(3(2y - \delta)\right) \leq 7/12 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large.

For Case 4 (c), if $x > z/2$, then the adversary generates three additional challenger jobs with value z for J_x. Then, the reward for A is at most $r(A) = 3z$. However, the optimal reward is $OPT = 3(x + z - \delta)$. Therefore, $\gamma_A = 3z/\left(3(x + z - \delta)\right) \leq 2/3 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large. Otherwise, if $x \leq z/2$, then the adversary generates three additional challenger jobs with value z for J_z. Then, the reward for A is at most $r(A) = 2x + 3z$. However, the optimal reward is $OPT = 3(2z - \delta)$. Therefore, $\gamma_A = (2x + 3z)/\left(3(2z - \delta)\right) \leq 2/3 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large.

For Case 4 (d), if $x > z/2$, then the adversary generates three additional challenger jobs with value z for J_x. Then, the reward for A is at most $r(A) = 3z$. However, the optimal reward is $OPT = 3(x + z - \delta)$. Therefore, $\gamma_A = 3z/\left(3(x + z - \delta)\right) \leq 2/3 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large. Otherwise, if $x \leq z/2$ and $y > 0.676z$, then the adversary generates three additional challenger jobs with value z for J_y. Then, the reward for A is at most $r(A) = x + 3z$. However, the optimal reward is $OPT = 3(y + z - \delta)$. Therefore, $\gamma_A = (x + 3z)/\left(3(y + z - \delta)\right) \leq 0.697 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large. Otherwise, if $x \leq z/2$ and $y \leq 0.676z$, then the adversary generates three additional challenger jobs with value z for J_z. Then, the reward for A is at most $r(A) = x + y + 3z$. However, the optimal reward is $OPT = 3(2z - \delta)$. Therefore, $\gamma_A = (x + y + 3z)/\left(3(2z - \delta)\right) \leq 0.696 + \zeta$, for δ sufficiently small and $\bar{v}$ sufficiently large.

Summarizing all the possible cases, since ζ can be arbitrarily small, no deterministic algorithm can achieve a competitive ratio lower than $1/0.696 = 1.436$ on three machines for C-benevolent job sequences versus adaptive adversaries. ■

CHAPTER 4

SCHEDULING C-BENEVOLENT JOBS ON WEIGHTED MACHINES

This chapter considers scheduling C-benevolent jobs on multiple heterogeneous machines with different positive weights. The reward for completing a job assigned to a machine is given by the product of the job value and the machine weight. The objective of this scheduling problem is to maximize the total reward for completed jobs. Two classes of approximation algorithms are analyzed, *Cooperative Greedy* algorithms and *Prioritized Greedy* algorithms, with competitive ratios provided. We show that when the weight ratios between machines are small, the *Cooperative Greedy* algorithm outperforms the *Prioritized Greedy* algorithm. As the weight ratios increase, the *Prioritized Greedy* algorithm outperforms the *Cooperative Greedy* algorithm. Moreover, as the weight ratios approach infinity, the competitive ratio of the *Prioritized Greedy* algorithm approaches four. We also provide lower bounds of 2 and 9/7 for the competitive ratio of any deterministic algorithm for scheduling C-benevolent jobs on two and three machines with arbitrary weights, respectively.

4.1 *Cooperative Greedy* Algorithms

This section considers the *Cooperative Greedy* algorithms, proposed in Chapter 3. First, we show that the performance of the *Cooperative Greedy* algorithm for two unweighted machines deteriorates significantly as the ratio of the two machine weights (referred to as the *weight ratio*) increases. Then, we extend this analysis to the *Pairing-m* algorithm on m machines (an extension of the *Cooperative Greedy* algorithm to multiple machines) and show that the competitive ratio increases as the largest weight ratio increases for even number of machines and is infinite for odd number of machines.

4.1.1 The *Cooperative Greedy* algorithm on two weighted machines

Without loss of generality, suppose $w_1 \geq w_2$. The *Cooperative Greedy* algorithm is 2-competitive for scheduling C-benevolent jobs on two unweighted machines. Theorem 18 shows that the competitive ratio of the *Cooperative Greedy* algorithm on two weighted machines increases as the weight ratio increases. Denote the weight ratio by $\beta = w_1/w_2$, and hence $\beta \geq 1$.

Theorem 18. *The Cooperative Greedy algorithm achieves a competitive ratio of* $(\beta + 1)$ *for scheduling C-benevolent jobs on two weighted machines, where* β *is the weight ratio between the two machines.*

Proof: We prove this by computing the lower and upper bounds for the competitive ratio of the *Cooperative Greedy* algorithm, which turn out to be the same value. The lower bound is given by considering a sequential game between an adaptive adversary and the algorithm. The upper bound is given by considering the performance of the *Cooperative Greedy* on two unweighted machines.

First, we consider the lower bound. Note that it is straightforward for the adversary to observe which machine is the primary machine (PM) using the following strategy. The adversary can send out two jobs $\{J_1', J_2'\}$ of identical value, J_2' immediately following J_1', and the algorithm will schedule them both. Then, the adversary sends out another job J_3' immediately following the arrival of job J_2', with $v(J_3') > v(J_2')$. The secondary machine (SM) will abort the previously assigned job in favor of job J_3', and hence the adversary will learn which machine is the primary machine. This is referred to as the *learning phase*. Since the values of these three jobs can be made arbitrarily small, we ignore the rewards from such a learning phase and assume the adversary knows which machine is the primary machine at the beginning of the sequential game.

The adversary's strategy depends on two cases: (a) M_1 is the primary machine, and (b) M_2 is the primary machine. For case (a), the adversary generates three jobs, $\mathbf{I}_1 = \{J_1 = (0, \delta v),\ J_2 = (\epsilon, v),\ J_3 = (2\epsilon, v)\}$ (in the following, we use a vector of two elements to represent a job $J_i = (a_i, v_i)$ and skip the dimension of the length, since the length and the job value are subject to some fixed C-benevolent function). In this case, δv is chosen to be sufficiently small compared to v, and ϵ is chosen to be sufficiently small such

that all the three jobs conflict with each other. Then, the *Cooperative Greedy* algorithm will schedule J_1 on M_1 and J_2 on M_2, with the reward obtained given by $R(\mathbf{I}_1) = w_1\delta v + w_2 v$. However, the optimal schedule should be scheduling J_2 on M_1 and J_3 on M_2, and hence the optimal reward is given by $OPT(\mathbf{I}_1) = (w_1 + w_2)v$. Therefore, the competitive ratio can be made arbitrarily close to $(\beta + 1)$ by setting δv arbitrarily small and v arbitrarily large. For case (b), the adversary generates two jobs, $\mathbf{I}_2 = \{J_1 = (0, v),\ J_2 = (\epsilon, \delta v)\}$. The choices of ϵ and δv are the same as in case (a). In this case, the *Cooperative Greedy* algorithm will schedule J_1 on M_2 and J_2 on M_1, with the reward obtained given by $R(\mathbf{I}_2) = w_1\delta v + w_2 v$. However, the optimal schedule should be scheduling J_1 on M_1 and J_2 on M_2, and hence the optimal reward is given by $OPT(\mathbf{I}_2) = w_1 v + w_2\delta v$. Therefore, the competitive ratio can be made arbitrarily close to β by setting δv arbitrarily small and v arbitrarily large. Therefore, the lower bound for the competitive ratio is $(\beta + 1)$.

Next, we consider the upper bound. Let OPT_1 denote the reward for the optimal schedule on a single machine. Let OPT_1^w and OPT_2^w denote the reward for the optimal schedule on two weighted machines, with $OPT_1 \geq OPT_2$. Therefore, $OPT_1 \geq OPT_1^w \geq OPT_2^w$. Let ALG_1 and ALG_2 denote the reward for the schedule on machine M_1 and M_2 under the *Cooperative Greedy* algorithm, respectively. Then $ALG_1 + ALG_2 \geq OPT_1$. Let γ denote the competitive ratio of the Cooperative Greedy algorithm. Therefore,

$$\gamma = \frac{w_1 OPT_1^w + w_2 OPT_2^w}{w_1 ALG_1 + w_2 ALG_2} \leq \frac{(w_1 + w_2)OPT_1}{w_2 OPT_1} = \beta + 1.$$

Since the lower bound and the upper bound assume the same value, then the competitive ratio for the *Cooperative Greedy* algorithm is $(\beta + 1)$ on two weighted machines. ■

4.1.2 The *Pairing-m* algorithm on multiple weighted machines

This section considers the *Pairing-m* algorithm for scheduling C-benevolent jobs on multiple weighted machines. The *Pairing-m* algorithm is a generalization from the *Cooperative Greedy* algorithm on two machines and hence is classified as a *Cooperative Greedy* algorithm for multiple machines.

We restate the *Pairing-m* algorithm. The *Pairing-m* algorithm is deter-

ministic for an even number of machines and randomized for an odd number of machines. For an even number of machines, $m/2$ of the m machines are arbitrarily selected as the PMs and the other machines are SMs initially. Jobs scheduled on PMs are always guaranteed to be completed, while jobs scheduled on SMs will be aborted if a new job arrives with a larger value. Whenever a job arrives, if there is some PM that is available, then the job is assigned to the PM. Otherwise, the job will be assigned to the SM with the job that has the smallest value among jobs on SMs if and only if the job value is greater than the smallest job value on SMs. Moreover, whenever a job on some PM is completed, the PM becomes a SM and the SM that is executing the job with the largest value among all SMs at that time becomes a PM. This role-switching (PM→SM and SM→PM) is repeated until all jobs are assigned. For an odd number of jobs, one virtual machine is added to make the total number of machines even ($m + 1$ real and virtual machines), and hence the *Pairing-(m+1)* algorithm for an even number of machines can be applied. However, since there are only m real machines, the schedule on each machine is arbitrarily selected with equal probability from schedules under the *Pairing-(m+1)* algorithm for $(m + 1)$ machines.

Without loss of generality, suppose that there are a total of m machines (M_1, M_2, …, M_m), with $w_1 \geq w_2 \geq \ldots \geq w_m$. Let $\beta_i = w_i/w_{i+1}$ denote the weight ratios for $i = 1, 2, \ldots, m-1$. Therefore, $\beta_i \geq 1$ for $i = 1, 2, \ldots, m-1$. Theorem 19 shows that the competitive ratio of the *Pairing-m* algorithm increases as the largest weight ratio increases, and hence the algorithm is not desirable when weight ratios are large.

Theorem 19. *The Pairing-m algorithm achieves a competitive ratio of* $O(w_1/w_m) = O(\Pi_{i=1}^{m-1}\beta_i)$ *for scheduling C-benevolent jobs on an even number of weighted machines with weight ratios* $\{\beta_i\}_{i=1}^{m-1}$. *However, no finite competitive ratio can be achieved for scheduling C-benevolent jobs on an odd number of machines.*

Proof: We prove this by considering the even and odd number of machine cases separately. Consider an even number of machines first. We will provide an upper bound $2w_1/w_m$ and a lower bound w_1/w_m for the competitive ratio of the *Pairing-m* algorithm.

Let OPT_i^w and OPT_i denote the reward for the optimal schedule on machine M_i for scheduling on weighted and unweighted machines, respectively,

for $i = 1, 2, \ldots, m$. Let $OPT^w_{(i)}$ ($OPT_{(i)}$) denote the ordering statistics for $\{OPT^w_i\}$ ($\{OPT_i\}$), indexed in decreasing order of rewards. Therefore, by the feasibility and optimality of these two optimal schedules,

$$\sum_{i=1}^{m} OPT^w_i \leq \sum_{i=1}^{m} OPT_i,$$

and

$$w_1 \sum_{i=1}^{m} OPT_i \geq \sum_{i=1}^{m} w_i OPT^w_i \geq \sum_{i=1}^{m} w_i OPT_i.$$

Let A_i denote the reward on machine M_i under the *Pairing-m* algorithm, for $i = 1, 2, \ldots, m$. Then, from Theorem 14 in Chapter 3,

$$\sum_{i=1}^{m} A_i \geq \sum_{i=1}^{m/2} OPT_{(i)} \geq 1/2 \sum_{i=1}^{m} OPT_i.$$

Therefore,

$$\begin{aligned}
\sum_{i=1}^{m} w_i A_i &\geq w_m \sum_{i=1}^{m} A_i \\
&\geq w_m/2 \sum_{i=1}^{m} OPT_i \\
&= \frac{w_m}{2w_1}(w_1 \sum_{i=1}^{m} OPT_i) \\
&\geq \frac{w_m}{2w_1}(\sum_{i=1}^{m} w_i OPT^w_i).
\end{aligned}$$

Therefore, the upper bound for the competitive ratio of the *Pairing-m* algorithm is $2w_1/w_m$.

Now consider an adversary playing against the *Pairing-m* algorithm by generating the sequence of jobs adaptively and trying to drive the competitive ratio of the algorithm as high as possible. We assume the adversary knows the strategy of the *Pairing-m* algorithm and so can straightforwardly tell which machines are the PMs at any time (the same as the learning phase in the proof of Theorem 18). Then the adversary follows a strategy that consists of only two kinds of jobs: (1) job J_1 with value δv and (2) job J_2

with value v, with $\delta v \ll v$. If machine M_m is the only available SM, the adversary generates a job J_2; otherwise, the adversary keeps generating job J_1 until all the other machines (M_1 to M_{m-1}) are busy with assigned jobs. Let n_1 and n_2 denote the total number of J_1 and J_2 jobs generated by the adversary. The adversary stops when machine M_m completes the first job J_2. Therefore, $n_2 = 1$ and $n_1 \leq m - 1$. Since the adversary only generates jobs when there is an available machine, all jobs generated by the adversary can be completed by the *Pairing-m* algorithm. Therefore, the reward for the *Pairing-m* algorithm is at most $n_1 w_1 \delta v + n_2 w_m v$. However, the optimal schedule assigns all J_2 jobs to machine M_1, and hence the optimal reward is at least $n_1 w_m \delta v + n_2 w_1 v$. Therefore, the competitive ratio of the *Pairing-m* algorithm can be driven arbitrarily close to w_1/w_m, with δv arbitrary small. Therefore, the lower bound for the competitive ratio of the *Pairing-m* algorithm is w_1/w_m.

Combining the upper and lower bound together, the competitive ratio of the *Pairing-m* algorithm is $O(w_1/w_m)$ for an even number of machines.

As for odd number of machines, the adversary follows the same strategy as above for $(m+1)$ (even number) machines and treats the virtual machine as a machine with weight zero (the lowest weight). Therefore, job J_2 will be assigned to the virtual machine, and hence, its value will be lost. The reward for the *Pairing-m* algorithm is at most $n_1 w_1 \delta v$. However, the optimal schedule assigns job J_2 to machine M_1, and hence the optimal reward is at least $n_2 w_1 v$. Therefore, no finite competitive ratio can be achieved for an odd number of machines. ■

4.2 *Prioritized Greedy* Algorithms

Competitive ratios of the *Cooperative Greedy* algorithm and the *Pairing-m* algorithm increase as the weight ratios increase, and hence the algorithms are not desirable when weight ratios are large. We study a *Prioritized Greedy* algorithm that uses a *Greedy* algorithm on each machine to achieve a better competitive ratio for large weight ratios. A Greedy algorithm with *abortion ratio* α (referred to as the Greedy-α algorithm) aborts a job J that is being executed in favor of a newly arrived job J' if and only if $v(J') > \alpha v(J)$, where $\alpha > 1$ (when a machine is not executing any job, we treat it as executing

a virtual job with value zero and length zero, which is compatible with C-benevolent jobs).

4.2.1 Two weighted machines

We consider the *Prioritized Greedy-*α_1, α_2 algorithm on two weighted machines as the base case, with $\alpha_1/(1-\alpha_1^{-1}) \leq \alpha_2/(1-\alpha_2^{-1})$. We do not specify the values of the two parameters, α_1 and α_2, at this point, which will be determined by optimizing the competitive ratio of the algorithm.

The *Prioritized Greedy-*α_1, α_2 algorithm is similar to employing a Greedy-

Algorithm 3 *Prioritized Greedy-*α_1, α_2 Algorithm

Let $S_1(t)$ and $S_2(t)$ denote the jobs being executed on machine M_1 and M_2 at time t, respectively.
for all job intervals J_i in an instance **I do**
  if $v_i > \alpha_1 v(S_1(a_i))$ **then**
    Abort $S_1(t)$ and assign J_i to machine M_1.
    Update $S_1(t)$, i.e., $S_1(t) = J_i$, for $t \in [a_i, a_i + l_i)$.
  else if $v_i > \alpha_2 v(S_2(a_i))$ **then**
    Abort $S_2(t)$ and assign J_i to machine M_2.
    Update $S_2(t)$, i.e., $S_2(t) = J_i$, for $t \in [a_i, a_i + l_i)$.
  else
    Discard J_i.
  end if
end for

α_1 algorithm on machine M_1 and a Greedy-α_2 algorithm on machine M_2. However, the analysis is more complicated than simply combining these two Greedy algorithms due to the priorities of the machines. Consider a job J_i such that both $v_i > \alpha_1 v(S_1(a_i))$ and $v_i > \alpha_2 v(S_2(a_i))$ hold. Then according to the *Prioritized Greedy-*α_1, α_2 algorithm, J_i will be assigned to M_1 rather than M_2. Therefore, those jobs assigned to M_1 (including both those completed on M_1 or temporarily assigned and later aborted on M_1) cannot go to M_2, and hence the Greedy-α_2 algorithm on M_2 is only influencing a subset of the job instance **I**. These issues need to be addressed in the analysis of the competitive ratio of the *Prioritized Greedy-*α_1, α_2 algorithm.

We will use a result for scheduling C-benevolent jobs on a single machine using Greedy-α algorithm given by [39]. We restate this result in Proposition 3.

Proposition 3. *[39, Remark 3.3] The Greedy-α algorithm achieves a competitive ratio of $\alpha/(1-\alpha^{-1})$ on a single machine for C-benevolent jobs, for $\alpha > 1$. Therefore, the Greedy-2 algorithm achieves a competitive ratio of 4 on a single machine for C-benevolent jobs, which is the best achievable competitive ratio for deterministic algorithms on a single machine.*

We refer to jobs that arrive during the execution of job J as *testing jobs* for J. For the *Greedy-α* algorithm, a job that is being executed will be aborted by the first testing job for it with a value greater than α times its value; a job that is being executed will be completed if none of the testing jobs for it has a value greater than α times its value. Therefore, values of subsequent jobs where successors abort previous jobs form a geometric sequence, and we will compute an upper bound for the optimal reward using this property of *Greedy* algorithms. A *geometric subsequence* is a sequence of jobs where: (1) each job (except the last two jobs) is aborted by its successive job; (2) the second to last job is the only completed job in the subsequence; (3) the last job is not assigned by the *Greedy-α* algorithm but has the largest completion time among all the testing jobs for the completed job. Observation 1 gives a lower bound for the reward for *Greedy* algorithms and an upper bound for the optimal reward, which is due to the convexity of C-benevolent jobs and is important in the analysis of *Prioritized Greedy* algorithms on multiple weighted machines. The proof of Observation 1 is identical to the proof of Proposition 3 and hence omitted here.

Observation 1. *For jobs arriving during the execution of a geometric subsequence, the Greedy-α algorithm achieves a $(1-\alpha^{-1})/\alpha$ fraction of the total value of all jobs in this geometric subsequence. In other words, the total value of all jobs in this subsequence is an upper bound for the optimal reward for jobs arriving during the execution of the subsequence.*

Let O_1 and O_2 denote the optimal schedule on machine M_1 and M_2 for an instance $\mathbf{I}$, respectively. Let $\tilde{O}_1$ denote the single-machine optimal schedule on M_1 for instance $\mathbf{I}$. Let A_1 denote the set of jobs assigned to M_1 under the Greedy-α_1 algorithm on M_1, either completed or aborted. Let G_1 denote the set of geometric subsequences generated by A_1 (by including the last testing job with the largest completion time for each geometric subsequence). Let $\tilde{O}_2$ denote the single-machine optimal schedule on M_2 for the residual jobs

$\mathbf{I}_r = \mathbf{I} \setminus A_1$. Let G_2 denote the set of geometric subsequences generated by jobs assigned on M_2. For any set of jobs S, let $v(S)$ denote the value of jobs contained in set S. Then from Observation 1, $v(G_1) \geq v(\tilde{O}_1)$ and $v(G_2) \geq v(\tilde{O}_2)$. Theorem 20 gives the competitive ratio of the *Prioritized Greedy-*α_1, α_2 algorithm.

Theorem 20. *The Prioritized Greedy-*α_1, α_2 *algorithm achieves a competitive ratio of* $4(1 + \frac{1}{\beta+1})$ *for scheduling C-benevolent jobs on two weighted machines, where* $\alpha_1 = 2, \alpha_2 = 2$ *and* $\beta = w_1/w_2 \geq 1$.

Proof: From the definitions of $\{O_1, O_2\}$ and $\{\tilde{O}_1, \tilde{O}_2\}$,

$$v(O_1) \leq v(\tilde{O}_1) \text{ and } v(O_2) \geq v(\tilde{O}_2),$$

which are due to the optimality of $\{O_1, O_2\}$ and the sequential optimality of $\tilde{O}_1$ and $\tilde{O}_2$. From Observation 1 and Proposition 1, the Greedy algorithm on M_1 will achieve at least a $(1 - \alpha_1^{-1})/\alpha_1$ fraction of $v(G_1)$ and a $(1 - \alpha_2^{-1})/\alpha_2$ fraction of $v(G_2)$. We want to show $w_1 v(G_1) + 2w_2 v(G_2) \geq w_1 v(O_1) + w_2 v(O_2)$.

We consider the difference between O_2 and $\tilde{O}_2$. If $A_1 \bigcap O_2 = \emptyset$, then $v(O_2) \leq v(\tilde{O}_2)$ since O_2 is a feasible schedule on M_2. Therefore, $v(O_2) = v(\tilde{O}_2)$ and

$$w_1 v(O_1) + w_2 v(O_2) \leq w_1 v(\tilde{O}_1) + w_2 v(\tilde{O}_2) \leq w_1 v(G_1) + w_2 v(G_2).$$

If $A_1 \bigcap O_2 \neq \emptyset$, then for any job $J_i \in A_1 \bigcap O_2$, there exists a set of jobs $\{J_j^i\}_{j=1}^h \subset O_1$ that conflict with J_i. Otherwise, J_i should be added to O_1 to obtain a larger reward for $w_1 v(O_1) + w_2 v(O_2)$ (since $w_1 \geq w_2$), which contradicts the optimality of $\{O_1, O_2\}$. First, we assume $\{J_j^i\}_{j=1}^h$ do not conflict with other jobs in $A_1 \bigcap O_2$. We will discuss the case where $\{J_j^i\}_{j=1}^h$ conflict with other jobs in $A_1 \bigcap O_2$ later.

Since J_i is assigned to M_1 under the *Greedy-*α_1 algorithm, all jobs $\{J_j^i\}_{j=1}^h$ can no longer be completed on M_1. However, since the *Greedy-*α_1 is a preemptive algorithm, then (a) the very first job, J_1^i, may be aborted by J_i on M_1, and (b) the very last job, J_h^i, may abort J_i on M_1. If both (a) and (b) occur, then $\{J_j^i\}_{j=2}^{h-1}$ will be available to machine M_2. Note that $\{J_j^i\}_{j=2}^{h-1}$ all arrive after J_i and can be completed before the completion time of job J_i, with $\sum_{j=2}^{h-1} v(J_j^i) \leq v(J_i)$. Therefore, they can be added to $O_2 \setminus J_i$ with no

conflict, and hence

$$\begin{aligned}
&w_1 v(G_1(\{J_1^i, J_i, J_h^i\})) + w_2 v(G_2(\{J_j^i\}_{j=2}^{h-1})) \\
\geq &w_1(v(J_i) + v(J_1^i) + v(J_h^i)) + w_2 \sum_{j=2}^{h-1} v(J_j^i) \\
\geq &w_1 \sum_{j=1}^{h} v(J_j^i) + w_2 v(J_i),
\end{aligned}$$

where $G_1(S)$ and $G_2(S)$ denote jobs in G_1 and G_2 that conflict with jobs in set S, respectively, with $G_1(\{J_1^i, J_i, J_h^i\}) = \{J_1^i, J_i, J_h^i\}$.

If (a) occurs and (b) does not occur, then $\{J_j^i\}_{j=2}^h$ will be available to machine M_2. Then, $\sum_{j=1}^h v(J_j^i) \leq v(G_1(\{J_j^i\}_{j=1}^h))$ and $\sum_{j=2}^h v(J_j^i) \leq v(G_2(\{J_j^i\}_{j=2}^h))$. Note that $J_1^i \in G_1(\{J_j^i\}_{j=1}^h)$ and $J_i \in G_1(\{J_j^i\}_{j=1}^h)$. If there is no job in O_2 conflicting with $\{J_j^i\}_{j=2}^h$, then $\{J_j^i\}_{j=2}^h$ can be added $O_2 \setminus J_i$ with no conflict: (a) if $\sum_{j=2}^h v(J_j^i) \geq v(J_i)$, then

$$w_1 v(G_1(\{J_j^i\}_{j=1}^h)) + w_2 v(G_2(\{J_j^i\}_{j=2}^h)) \geq w_1 \sum_{j=1}^{h} v(J_j^i) + w_2 v(J_i);$$

(b) if $\sum_{j=2}^h v(J_j^i) \leq v(J_i)$, then

$$w_1 v(J_1^i) + w_1 v(J_i) + w_2 \sum_{j=2}^{h} v(J_j^i) \geq w_1 \sum_{j=1}^{h} v(J_j^i) + w_2 v(J_i),$$

which follows from Hardy's lemma, and hence

$$w_1 v(G_1(\{J_j^i\}_{j=1}^h)) + w_2 v(G_2(\{J_j^i\}_{j=2}^h)) \geq w_1 \sum_{j=1}^{h} v(J_j^i) + w_2 v(J_i).$$

If there are jobs in O_2 conflicting with $\{J_j^i\}_{j=2}^h$, denote this set of jobs by $\{J_n\}$. From our assumption, then $\{J_n\} \not\subset A_1$. Then

$$\sum_{j=2}^{h} v(J_j^i) + \sum_{n} v(J_n) \leq 2v(G_2(\{J_j^i\}_{j=2}^h \bigcup \{J_n\})),$$

which follows from Observation 1. Therefore,

$$\begin{aligned} & w_1 v(G_1(\{J_j^i\}_{j=1}^h)) + 2w_2 v(G_2(\{J_j^i\}_{j=2}^h \bigcup J_n)) \\ \geq & w_1 \sum_{j=1}^{h} v(J_j^i) + w_2(v(J_i) + v(J_n)). \end{aligned}$$

If (a) does not occur and (b) occurs, then $\{J_j^i\}_{j=1}^{h-1}$ will be available to machine M_2. The arguments are identical as those for the case of (a) occurs and (b) does not occur by substituting $\{J_j^i\}_{j=2}^h$ with $\{J_j^i\}_{j=1}^{h-1}$.

If neither (a) nor (b) occurs, then $\{J_j^i\}_{j=1}^h$ will be available to machine M_2. The arguments are identical to those for the case of (a) occurs and (b) does not occur by substituting $\{J_j^i\}_{j=2}^h$ with $\{J_j^i\}_{j=1}^h$.

If $\{J_j^i\}_{j=1}^h$ conflict with other jobs in $A_1 \bigcap O_2$, then we use the same arguments to analyze these jobs together. That is, consider a segment of O_2, denoted by Q_i, with $Q_i \bigcap A_1 \neq \emptyset$ and $O_1(Q_i) \bigcap (O_2 \setminus Q_i) = \emptyset$, where $O_1(Q_i)$ denote the set of jobs in O_1 that conflict with Q_i. Identical arguments can be applied by substituting J_i with Q_i.

Therefore, we have shown that $w_1 v(G_1) + 2w_2 v(G_2) \geq w_1 v(O_1) + w_2 v(O_2)$. Let D_1 and D_2 denote the set of jobs completed by M_1 and M_2, respectively. Setting $\alpha_1 = 2$ (to maximize the value of $v(D_1)$) leads to

$$\begin{aligned} v(D_1) &\geq \frac{\alpha_1 - 1}{\alpha_1^2} v(G_1) \geq \frac{1}{4} v(O_1), \\ v(D_2) &\geq \frac{\alpha_2 - 1}{\alpha_2^2} v(G_2), \end{aligned} \tag{4.1}$$

Therefore, the reward for the *Prioritized Greedy*-$2, \alpha_2$ algorithm is

$$w_1 v(D_1) + w_2 v(D_2) \geq \frac{1}{4} w_1 v(G_1) + \frac{\alpha_2 - 1}{\alpha_2^2} w_2 v(G_2).$$

Let γ denote the competitive ratio of the *Prioritized Greedy*-$2, 2$ algorithm. Then

$$\begin{aligned} \gamma = \frac{w_1 v(O_1) + w_2 v(O_2)}{w_1 v(D_1) + w_2 v(D_2)} &\leq \frac{w_1 v(G_1) + 2w_2 v(G_2)}{\frac{1}{4} w_1 v(G_1) + \frac{\alpha_2 - 1}{\alpha_2^2} w_2 v(G_2)} \\ &\leq \frac{\beta + 2}{\frac{1}{4}\beta + \frac{\alpha_2 - 1}{\alpha_2^2}}, \end{aligned}$$

which is minimized by taking $\alpha_2 = 2$, and hence $\gamma = 4(1 + \frac{1}{\beta+1})$. ■

From Theorem 20, the competitive ratio of the *Prioritized Greedy*-2, 2 approaches a constant 4 as the weight ratio β approaches infinity. Therefore, when β is small (i.e., $\beta \leq 3.828$), the Cooperative Greedy algorithm has a smaller competitive ratio than the *Prioritized Greedy*-2, 2 algorithm. However, as β becomes larger (i.e., $\beta > 3.828$), the *Prioritized Greedy*-2, 2 algorithm becomes better than the Cooperative Greedy algorithm.

4.2.2 Multiple weighted machines

This section derives the competitive ratio of the *Prioritized Greedy* algorithm on m weighted machines (M_1, $M_2, \ldots$, M_m), with $w_1 \geq w_2 \geq \ldots \geq w_m$. Each machine employs a Greedy algorithm, and the priorities for machines M_1, $M_2, \ldots$, M_m go from the highest to the lowest. Whenever a job arrives, it goes to the machine with the highest priority to be assigned; if it is not assigned on this machine, then it goes to the machine with the next highest priority; if it is not assigned on the machine with the lowest priority, then the job is discarded.

We provide the competitive ratio of the *Prioritized Greedy* algorithm on multiple machines by considering the relationship between the optimal schedule and the *sequentially achievable optimal schedule*. Let O_i denote the optimal schedule on machine M_i for job instance $\mathbf{I}$, for $i = 1, 2, \ldots, m$. Let A_i denote the set of jobs assigned to machine M_i, either completed or aborted, for $i = 1, 2, \ldots, m$. Therefore, $\{A_i\}_{i=1}^m$ are non-overlapping from each other. Let G_i denote the set of geometric subsequences generated by jobs assigned on machine M_i, for for $i = 1, 2, \ldots, m$. Let $\tilde{O}_i$ denote the sequentially achievable optimal schedule on machine M_i, which is defined as the optimal schedule for jobs $\mathbf{I}_i \triangleq \mathbf{I} \setminus \bigcup_{k=1}^{i-1} A_k$ with $\bigcup_{k=1}^{0} A_k \triangleq \emptyset$, for $i = 1, 2, \ldots, m$. Then from Observation 1, $v(G_i) \geq v(\tilde{O}_i)$ by definition. Theorem 21 shows the relationship between the reward for the optimal schedule and the upper bound for the sequentially achievable optimal schedule.

Theorem 21. *Let $\{O_i\}_{i=1}^m$ denote the optimal schedule on the m weighted machines. Then for C-benevolent jobs:*

$$\sum_{i=1}^{m} w_i v(O_i) \leq (1 + \frac{1}{\min_i \beta_i}) \sum_{i=1}^{m} w_i v(G_i),$$

where $\beta_i = w_i/w_{i+1}$ *for* $i = 1, 2, \ldots, m-1$, *with* $w_1 \geq w_2 \ldots \geq w_m$.

Proof: We prove this by induction on m. First, consider the base case, $m = 2$. Then

$$\begin{aligned} w_1 v(O_1) + w_2 v(O_2) &\leq (w_1 + w_2) v(O_1) \\ &\leq w_1(1 + 1/\beta) v(\tilde{O}_1) \\ &\leq w_1(1 + 1/\beta) v(G_1), \end{aligned}$$

where $v(G_1) \geq v(\tilde{O}_1) \geq v(O_1)$, and hence Theorem 21 holds trivially for $m = 2$.

Suppose Theorem 21 holds for $(m-1)$ machines, where $m \geq 3$. We prove it holds for m machines. If $\tilde{O}_m = O_m$, then the result follows from $v(G_m) \geq v(\tilde{O}_m)$ and the induction assumption. Otherwise, consider the set of jobs $O_m^r = \{J : J \in O_m, J \notin \bigcup_{i=1}^{m-1} A_i\}$. Then $v(O_m^r) \leq v(\tilde{O}_m) \leq v(G_m)$ due to the sequential optimality of $\tilde{O}_m$.

Consider a job $J_1 \in A_{i*} \bigcap O_m$ for some $i* = 1, 2, \ldots, m-1$. Then $J_1 \notin \tilde{O}_m$ and $J_1 \in G_{i*}$. Therefore,

$$w_m v(J_1) = \frac{1}{\beta*} w_{i*} v(J_1) \leq \frac{1}{\min_k \beta_k} w_{i*} v(J_1),$$

where $\beta* = w_{i*}/w_m$. In this case, J_1 is said to be *charged*. Whenever a job $J_j \in G_i$ is charged for $i = 1, 2, \ldots, m-1$, an additional reward of $1/\min_k \beta_k w_i v(J_j)$ will be added to the total reward of $\sum_{i=1}^m w_i v(G_i)$. Note that J_1 can be charged by any job belonging to optimal schedules on machines with lower priorities at most once (since $\{O_i\}_{i=1}^m$ are non-overlapping). Therefore, Theorem 21 holds for m machines, which completes the proof.

■

Theorem 22 provides the competitive ratio of the *Prioritized Greedy* algorithm for m weighted machines, with the abortion ratios for all machines set to be two.

Theorem 22. *The competitive ratio of the Prioritized Greedy algorithm is* $4(1 + \frac{1}{\min_i \beta_i})$ *when all the Greedy algorithms on* m *machines set the abortion ratio to be* $\alpha_i = 2$ *for* $i = 1, 2, \ldots, m$.

Proof: Let D_i denote the set of jobs completed on machine M_i under the

Prioritized Greedy algorithm, for $i = 1, 2, \ldots, m$. Then from Observation 1,

$$\sum_{i=1}^{m} w_i v(D_i) \geq \frac{1}{4} \sum_{i=1}^{m} w_1 v(G_i),$$

where all the *Greedy* algorithms set the abortion ratio as $\alpha = 2$. Let γ_m denote the competitive ratio of the *Prioritized Greedy* algorithm for m machines. Then from Theorem 21,

$$\begin{aligned} \gamma = \frac{\sum_{i=1}^{m} w_i v(O_i)}{\sum_{i=1}^{m} w_i v(D_i)} &\leq 4 \frac{(1 + \frac{1}{\min_i \beta_i}) \sum_{i=1}^{m} w_i v(\tilde{O}_i)}{\sum_{i=1}^{m} w_i v(\tilde{O}_i)} \\ &= 4(1 + \frac{1}{\min_i \beta_i}). \end{aligned}$$

■

From Theorem 22, the competitive ratio for two weighted machines is $4(1+1/\beta)$, where β is the weight ratio. This is a weaker result than Theorem 20, which gives a competitive ratio of $4(1 + 1/(\beta + 1))$.

4.3 Lower Bounds for Competitive Ratios of Deterministic Algorithms

This section provides lower bounds for the competitive ratio of any deterministic algorithm for scheduling C-benevolent job sequences on two and three weighted machines, respectively. These two constant lower bounds hold for arbitrary machine weights. We use an approach similar to that presented in [40] for the unweighted machine case. We use the *W-set* defined in Section 3.5. An example of W-sets can be found in Figure 4.1.

Theorem 23 provides a lower bound for the competitive ratio of any deterministic algorithm for scheduling C-benevolent jobs on two weighted machines.

Theorem 23. *No deterministic algorithm for scheduling C-benevolent jobs on two weighted machines can achieve a competitive ratio less than 2.*

Proof: We prove this lower bound by considering a sequential game between a deterministic algorithm and an adaptive adversary. We will show that there is a strategy for the adversary to drive the competitive ratio of

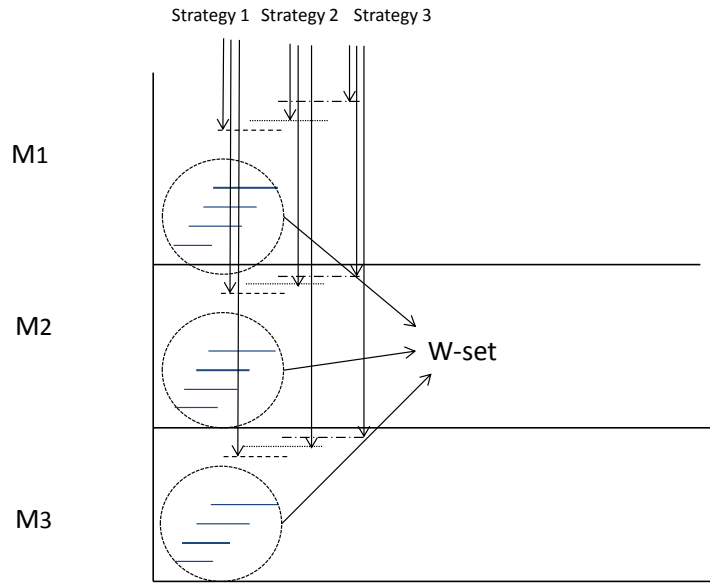

Figure 4.1: Adversary's strategy for three weighted machines using *W-sets*.

any deterministic algorithm to a value greater than $3/2 - \zeta$, where $\zeta > 0$ can be arbitrarily small.

First, the adversary generates two identical *W-sets*, with one set arriving slightly after the other with a delay of $\epsilon' \ll \epsilon$. We make this difference between these two sets only to conform to the assumption that no two jobs share the same arrival time. Since the delay between the two sets is sufficiently small compared to the difference between arrival times of subsequent jobs within each set, we will ignore this ϵ'. Now a deterministic algorithm, denoted by $\mathcal{A}$, has several choices of how to assign these jobs to each machine. Let $J_x(x)$ and $J_y(y)$ denote the jobs (job values) that $\mathcal{A}$ assigns to M_1 and M_2, respectively. Let OPT denote the optimal reward and $r(\mathcal{A})$ denote the reward for algorithm $\mathcal{A}$. Then the competitive ratio of $\mathcal{A}$ is $\gamma_A = OPT/r(\mathcal{A})$. The adversary will react adaptively to different choices made by $\mathcal{A}$. We divide the situation into different cases based on the values of x and y, as listed in Table 4.1.

1. **Case 1** $x = y = 1$. In this case, the adversary generates no more jobs. Therefore, $OPT = \bar{v}(w_1 + w_2)$, $r(\mathcal{A}) = w_1 + w_2$, and hence $1/\gamma_A = 1/\bar{v} \leq 1/2$, for $\bar{v} \geq 2$.

Table 4.1: All Cases for Two Weighted Machines

	Case 1	Case 2	Case 3(a)	Case 3(b)	Case 4(a)	Case 4(b)
x	1	> 1	1	> 1	> 1	> 1
y	1	x	$> x$	1	$> x$	$< x$

2. **Case 2** $x = y > 1$. In this case, the adversary generates two additional jobs that arrive right before the completion time of J_x and J_y but after the completion time of their preceding job in the W-set (jobs that arrive right before the completion time of some job J but after the completion time of the job preceding J in the W-set are referred to as the *challenger jobs* for J). The values of these two new jobs are both equal to x. Then, no matter whether $\mathcal{A}$ aborts x and y in favor of the new arrivals or not, the reward for $\mathcal{A}$ is at most $r(\mathcal{A}) = x(w_1 + w_2)$. However, the optimal reward is $OPT = (2x - \delta)(w_1 + w_2)$. Therefore, $1/\gamma_A = x/(2x - \delta) \leq 1/2 + \zeta$, for δ sufficiently small.

3. **Case 3 (a)** $1 = x < y$. In this case, the adversary will generate no more jobs. Therefore, $OPT = \bar{v}(w_1 + w_2)$, $r(\mathcal{A}) = w_1 + yw_2$, and hence

$$\begin{aligned} 1/\gamma_A &= (w_1 + yw_2)/(\bar{v}(w_1 + w_2)) \\ &\leq (w_1 + \bar{v}w_2)/(\bar{v}(w_1 + w_2)) \\ &\leq 1/2 + 1/(\bar{v} + 1) \leq 1/2 + \zeta, \end{aligned}$$

   for $\bar{v}$ sufficiently large.

4. **Case 3 (b)** $1 = y < x$. In this case, the adversary takes one of two possible strategies depending on the value of x. If $x \leq \bar{v}/2$, then the adversary generates no more jobs. Therefore, $OPT = \bar{v}(w_1 + w_2)$, $r(\mathcal{A}) = xw_1 + w_2$, and hence $1/\gamma_A = (xw_1 + w_2)/(\bar{v}(w_1 + w_2)) \leq 1/2$, for $\bar{v} \geq 2$. If $x > \bar{v}/2$, then the adversary generates two challenger jobs for J_x with value x. Then, no matter whether $\mathcal{A}$ aborts x in favor of the newly arrival or not, the reward for $\mathcal{A}$ is at most $r(\mathcal{A}) = xw_1 + (1+x)w_2$. However, the optimal reward is $OPT = (2x - \delta)(w_1 + w_2)$. Therefore,

$$\begin{aligned} 1/\gamma_A &= (xw_1 + (1 + x)w_2)/((2x - \delta)(w_1 + w_2)) \\ &\leq 1/2 + (1 + \delta)/(2x + 1) \leq 1/2 + \zeta, \end{aligned}$$

for $\bar{v}$ sufficiently large.

5. **Case 4 (a)** $1 < x < y$. In this case, the adversary takes a two-step strategy depending on the actions of $\mathcal{A}$. Note that if $x < y \leq \bar{v}/2$, then the adversary takes no more actions, and $1/\gamma_A \leq 1/2$. Therefore, suppose $y > \bar{v}/2$. In the first step, the adversary generates two challenger jobs for J_x with value y. If $\mathcal{A}$ does not abort J_x on M_1 in favor of the new arrival, the adversary takes no second step. Therefore, $OPT \geq (x - \delta + y)(w_1 + w_2)$ and $r(\mathcal{A}) = xw_1 + yw_2$, and hence $1/\gamma_A \leq (xw_1 + yw_2)/((x - \delta + y)(w_1 + w_2)) \leq 1/2 + \delta/(x + y) \leq 1/2 + \zeta$, for δ sufficiently small or $\bar{v}$ sufficiently large. Otherwise, J_x is aborted by $\mathcal{A}$ in favor of the new arrival, and the adversary takes the second step, where the adversary generates two challenger jobs for J_y with value y. Therefore, $OPT \geq (y - \delta + y)(w_1 + w_2)$, $r(\mathcal{A}) \leq yw_1 + yw_2$, and hence, $1/\gamma_A \leq 1/2 + \delta/y \leq 1/2 + \zeta$, for δ small enough or $\bar{v}$ large enough.

6. **Case 4 (b)** $1 < y < x$. In this case, the adversary takes one of two possible strategies depending on the relationship between x and y. If $y > x/2$, then the adversary generates two challenger jobs for J_y with value x. Therefore, $OPT \geq (x + y - \delta)(w_1 + w_2) \geq 3/2(x - \delta)(w_1 + w_2)$ and $r(\mathcal{A}) \leq x(w_1 + w_2)$, and hence, $1/\gamma \leq 2/3 + \zeta$. Alternatively, if $y \leq x/2$, then the adversary generates two challenger jobs for J_x with value x. Therefore, $OPT \geq (2x - \delta)(w_1 + w_2)$ and $r(\mathcal{A}) \leq x(w_1 + w_2) + yw_2 \leq x(w_1 + 3w_2/2)$, and hence, $1/\gamma \leq 2/3 + \zeta$.

Summarizing all the possible cases, since ζ can be arbitrarily small, no deterministic algorithm for C-benevolent jobs can achieve a competitive ratio less than 3/2 on two weighted machines. This upper bound does not depend on the weight ratio β.

Using similar techniques, Theorem 24 provides a lower bound for the competitive ratio of any deterministic algorithm for scheduling C-benevolent jobs on three weighted machines.

Theorem 24. *No deterministic algorithm for scheduling C-benevolent jobs on three weighted machines can achieve a competitive ratio less than 9/7.*

Proof: We establish this lower bound by considering a sequential game between a deterministic algorithm and an adaptive adversary. We will show

that there is a strategy for the adversary to drive the competitive ratio of any deterministic algorithm to $9/7-\zeta$, where $\zeta > 0$ can be arbitrarily small.

First, the adversary generates three identical W-sets, with sets arriving one set slightly after another by a delay of $\epsilon' \ll \epsilon$. This ϵ' is chosen to be sufficiently small such that any two jobs with identical values and only a difference of ϵ' in arrival times can be treated as identical jobs, same as in the Proof of Theorem 23. Now a deterministic algorithm, denoted by $\mathcal{A}$, has several choices of what jobs to assign. Let $J_x(x)$, $J_y(y)$, and $J_z(z)$ denote the job (value) that $\mathcal{A}$ assigns to M_1, M_2, and M_3, respectively. Let OPT denote the optimal reward and $r(\mathcal{A})$ denote the reward for algorithm $\mathcal{A}$. Then the competitive ratio of $\mathcal{A}$ is $\gamma_A = OPT/r(\mathcal{A})$. The adversary will react adaptively to different choices made by $\mathcal{A}$.

First, if some machine does not have any job assigned to it by $\mathcal{A}$, then the adversary treats that machine as being assigned the first job of the W-sets (the job with value one). This manipulation will only increase the reward for $\mathcal{A}$. Moreover, it allows us to classify actions by the adversary into different *scenarios* to simplify the proof, which can be tedious when enumerating all cases. Therefore, we assume that all three machines have been assigned one job from the W-sets by $\mathcal{A}$ in the following.

For three weighted machines, there are 16 cases in total considering the relationship between x, y, z and the number of value-one jobs in $\{x, y, z\}$ given $w_1 \geq w_2 \geq w_3$ (see Table 4.2). However, since strategies for the adversary are the same in some cases, we classify these cases into one *scenario* and provide a lower bound for each scenario. More precisely, we divide all cases into different scenarios based on the number of scheduled jobs with value one, as listed in Table 4.2. To avoid repeated statement, we set $\bar{v} > 2$ sufficiently large and ϵ sufficiently small.

1. **Scenario 1** All of the three scheduled jobs have a value of one. That is, $x = y = z = 1$. Then it is straightforward that $1/\gamma_A = 1/\bar{v} < 1/2$, for $\bar{v} > 2$.

2. **Scenario 2** Two of the scheduled jobs have a value of one. Consider the case $x > 1$ and $y = z = 1$. Then, if $x \leq \bar{v}/2$, the adversary generates no more jobs. Therefore, $r(\mathcal{A}) = xw_1 + w_2 + w_3$, $OPT = \bar{v}(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 1/2$. Otherwise, the adversary generates three challenger jobs for J_x with value x. Therefore, $r(\mathcal{A}) \leq$

Table 4.2: All Cases for Three Weighted Machines

Scenario	Cases
1	$x = y = z = 1$
2	$(x > 1, y = z = 1)$, $(y > 1, x = z = 1)$, $(z > 1, x = y = 1)$
3	$(x = 1, y \geq z), (x = 1, y < z)$, $(y = 1, x \geq z), (y = 1, x < z)$, $(z = 1, x \geq y), (z = 1, x < y)$
4	$(x \geq y \geq z > 1), (x \geq z \geq y > 1)$, $(y \geq x \geq z > 1), (y \geq z \geq x > 1)$, $(z \geq x \geq y > 1), (z \geq y \geq x > 1)$

$xw_1 + (w_2 + w_3)(1 + x)$, $OPT = (2x - \delta)(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 1/2 + \zeta$. The cases $y \neq 1$ and $z \neq 1$ follow the same strategy with $x(J_x)$ substituted by $y(J_y)$ and $z(J_z)$, respectively.

3. **Scenario 3** One of the scheduled jobs has a value of one. Consider the case $x \geq y > 1$ and $z = 1$. If $x \leq \bar{v}/2$, then the adversary generates no additional jobs, and $\gamma_A \leq 1/2$. Otherwise, if $x > \bar{v}/2$ and $y \leq x/2$, then the adversary generates three challenger jobs for J_x with value x. Therefore, $r(\mathcal{A}) \leq xw_1 + w_2(y + x) + w_3(1 + x)$, $OPT = (2x-\delta)(w_1+w_2+w_3)$, and hence $1/\gamma_A \leq 3/4+\zeta$. Alternatively, if $x > \bar{v}/2$ and $y > x/2$, then the adversary generates three challenger jobs for J_y with value x. Therefore, $r(\mathcal{A}) \leq xw_1 + w_2x + w_3(1 + x)$, $OPT = (x + y - \delta)(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 2/3 + \zeta$. For other cases in Scenario 3, the same strategy can be applied.

4. **Scenario 4** None of the scheduled jobs has a value of one. In this scenario, the adversary's strategy depends on which machine has been assigned the job with the largest value among all the three scheduled jobs, denoted by J_{max}. We provide lower bounds for the resulting three sub-scenarios: (a) J_{max} is assigned to machine M_1; (b) J_{max} is assigned to machine M_2; and (c) J_{max} is assigned to machine M_3.
   First, consider sub-scenario (a). Then, if $x \leq \bar{v}/2$, the adversary generates no additional jobs, and the competitive ratio is greater than 2. If $x > \bar{v}/2$, then there are two cases to be considered regarding the relationship between y and z: (1) $y \geq z$ and (2) $y \leq z$.

Consider case $x \geq y \geq z$. If $z > x/3$, then the adversary generates three challenger jobs for J_z with value x. Therefore, $r(\mathcal{A}) \leq x(w_1+w_2+w_3)$, $OPT = (x+z-\delta)(w_1+w_2+w_3)$, and hence $1/\gamma_A \leq 3/4+\zeta$. Otherwise, if $z \leq x/3$ and $y \geq x/2$, then the adversary generates three challenger jobs for J_y with value x. Therefore, $r(\mathcal{A}) \leq x(w_1 + w_2 + w_3) + zw_3 \leq x(w_1 + w_2 + 4w_3/3)$, $OPT = (x + y - \delta)(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 7/9 + \zeta$. Alternatively, if $z \leq x/3$ and $y \leq x/2$, then the adversary generates three challenger jobs for J_x with value x. Therefore, $r(\mathcal{A}) \leq x(w_1 + w_2 + w_3) + yw_2 + zw_3 \leq x(w_1 + 3w_2/2 + 4w_3/3)$, $OPT = (2x - \delta)(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 3/4 + \zeta$.
Consider case $x \geq z \geq y$. If $y > x/3$, then the adversary generates three challenger jobs for J_y with value x. Therefore, $r(\mathcal{A}) \leq x(w_1+w_2+w_3)$, $OPT = (x+y-\delta)(w_1+w_2+w_3)$, and hence $1/\gamma_A \leq 3/4+\zeta$. Otherwise, if $y \leq x/3$ and $z \geq x/2$, then the adversary generates three challenger jobs for J_z with value x. Therefore, $r(\mathcal{A}) \leq x(w_1 + w_2 + w_3) + yw_2 \leq x(w_1 + 4w_2/3 + w_3)$, $OPT = (x + z - \delta)(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 7/9 + \zeta$. Alternatively, if $y \leq x/3$ and $z \leq x/2$, then the adversary generates three challenger jobs for J_x with value x. Therefore, $r(\mathcal{A}) \leq x(w_1 + w_2 + w_3) + yw_2 + zw_3 \leq x(w_1 + 4w_2/3 + 3w_3/2)$, $OPT = (2x - \delta)(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 3/4 + \zeta$.
Consider sub-scenario (b) case $y \geq z \geq x$. If $x > y/3$, then the adversary generates three challenger jobs for J_x with value y. Therefore, $r(\mathcal{A}) \leq y(w_1 + w_2 + w_3)$, $OPT = (x + y - \delta)(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 3/4 + \zeta$. Otherwise, if $x \leq y/3$ and $z \geq 5y/7$, then the adversary generates three challenger jobs for J_z with value y. Therefore, $r(\mathcal{A}) \leq y(w_1 + w_2 + w_3) + xw_1 \leq x(4w_1/3 + w_2 + w_3)$, $OPT = (z + y - \delta)(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 7/9 + \zeta$. Alternatively, if $x \leq y/3$ and $z \leq 5y/7$, then the adversary generates three challenger jobs for J_y with value y. Therefore, $r(\mathcal{A}) \leq y(w_1+w_2+w_3)+ xw_1 + zw_3 \leq y(4w_1/3 + w_2 + 12w_3/7)$, $OPT = (2y - \delta)(w_1 + w_2 + w_3)$, and hence $1/\gamma_A \leq 7/9$. For case $y \geq x \geq z$, the strategy of the adversary is identical to case $x \geq y \geq z$ with $x(J_x)$ and $y(J_y)$ interchanged.
Consider sub-scenario (c). For case $z \geq x \geq y$, the strategy of the adversary is identical to case $y \geq x \geq z$ with $z(J_z)$ and $y(J_y)$ interchanged. For case $z \geq y \geq x$, the strategy of the adversary is identical to case $y \geq z \geq x$ with $z(J_z)$ and $y(J_y)$ interchanged.

Summarizing all the possible scenarios, since ζ can be arbitrarily small, no deterministic algorithm can achieve a competitive ratio less than 9/7 on three weighted machines for C-benevolent jobs. ■

4.4 Discussion

We compare our analysis results with the existing results for scheduling C-benevolent jobs on unweighted machines. From Chapter 3, the *Greedy-2* algorithm is 4-competitive for scheduling C-benevolent jobs on multiple unweighted jobs. The *Pairing-m* algorithm achieves competitive ratios of 2 and $2 + 2/m$ for an even and odd number of machines, respectively. Note that the unweighted machine case is a special case of the weighted machine case with $w_1 = 1$ for all $i = 1, 2, \ldots, m$. From Theorems 18 and 19, the *Cooperative Greedy* algorithm achieves a competitive ratio of 2 for two unweighted machines and the *Pairing-m* algorithm achieves a competitive ratio of $O(1)$ for an even number of unweighted machines, which are consistent with the results given in Chapter 3. As for the *Prioritized Greedy* algorithm, the *Prioritized Greedy-2,2* algorithm achieves competitive ratios of 6 and 8 for two and multiple unweighted machines, respectively, which are weaker than the *Greedy-2* algorithm. This is because: (1) the *Greedy-2* algorithm can be seen as treating the m machines as a single machine with capacity m, and hence the competitive ratio of the *Greedy-2* algorithm in [39] can be generalized to the *Greedy-2* algorithm for multiple machines; (2) we are dealing with the case of weighted machines with arbitrary weights and the expression for the competitive ratio given in this chapter holds for any machine weights, and hence the analysis is not necessarily tight for each weight ratio.

The results in this chapter focus on C-benevolent jobs. However, these results can be generalized to equal-length jobs. [47] proposes a 2-competitive algorithm for scheduling equal-length jobs on even number of unweighted machines, an idea similar to the *Cooperative Greedy* algorithm analyzed in this chapter. Our analysis techniques in Section 4.1 can be applied directly to show the competitive ratio of the *Cooperative Greedy* algorithm is $O(w_1/w_m)$ for even number of weighted machines and infinite for odd number of machines (if the randomization technique for odd number of machines proposed in Chapter 3 is used). For the *Prioritized Greedy* algorithm, the competitive

ratio analysis applies without any change. Similarly, the upper bounds of 2 and 9/7 for two and three weighted machines still hold for equal-length jobs.

The reward for some machine to complete an assigned job is assumed in this chapter to be the product of the machine weight and the job value. However, our results are not limited to this specific reward function. It is straightforward to see that our results can be generalized to any reward function with the form as the product of the machine weight and a nondecreasing convex function of the job value (since the C-benevolent function $v = g(l)$ is convex, then $g_1 \circ g$ is also convex if $g_1(x)$ is a nondecreasing convex function of x). Developing approximation algorithms for online interval scheduling problems with other reward functions is a future research direction.

The interval scheduling model assumes the length of a job to be the same no matter which machine it is assigned to. This assumption needs to be relaxed by introducing machine speeds, since machines with different weights usually have different speeds (higher screening levels have higher true alarm rates and typically take longer to screen a passenger). Therefore, the completion time of a job depends on the machine it is assigned to (the sum of the arrival time and the ratio of the length to the machine rate). There have been many research results on the interval scheduling problem on uniformly related machines [46] or unrelated machines [66]. Scheduling jobs on uniformly related weighted machines is a direction of current investigation.

CHAPTER 5

STOCHASTIC ONLINE INTERVAL SCHEDULING PROBLEMS

This chapter considers stochastic online interval scheduling problems, where a set of sequentially arriving jobs are to be matched to a group of workers. The objective is to maximize the total expected reward, defined as the sum of the rewards of each completed job. Each worker can be assigned to multiple jobs subject to the constraint that previously assigned jobs are completed. We provide 2-competitive online algorithms for independent and identically distributed equal-length and memoryless-length jobs. We also provide e-competitive and $e^2/(e-1)$-competitive online algorithms for jobs with a random order of arrivals for equal-length and memoryless-length jobs, respectively. We further show e^p-competitive and $e^{p+1}/(e-1)$-competitive online algorithms for equal-length and memoryless-length jobs, respectively, for both geometric with parameter p and a random order of arrivals.

5.1 Formulation

This section formulates the stochastic online interval scheduling problem as a sequential stochastic assignment problem with reusable workers (RSSAP). The model is described using the classic SSAP setting with workers and jobs, which may be extended to other scenarios including interval scheduling problems and online bidding problems.

Let m_0 denote the total number of workers and $\{w_m\}$ denote the success rate for each worker, with $w_1 \leq w_2 \leq \ldots \leq w_{m_0}$. A worker who is assigned to a job and has not completed that job is said to be *busy*; otherwise, the worker is said to be *available*. Each arriving job has a three-dimensional vector revealed upon arrival, with the first component representing the arrival time, the second component representing the job value and the third component representing the job length. For example, if a job length is 2 and a worker is

assigned to the job at time $t = 1$, the worker becomes available again at time $t = 3$. Let $\mathbf{J}_n = (a_n, v_n, l_n)$ denote the three-dimensional vector for the n^{th} job arrival, $n = 1, 2, \ldots$. Let $a(J)$, $v(J)$ an $l(J)$ to represent the arrival time, value and length of job J, respectively. We study two classes of independent and identically distributed (IID) job sequences:

> *Equal-length* job sequences: lengths of jobs are the same, independent of values of jobs;
>
> *Memoryless-length* job sequences:lengths of jobs follow a geometric distribution with parameter q, independent of values of jobs.

Let $F_v(v)$ denote the cdf for job values.

In RSSAP, the time axis is divided into *slots*, indexed by $t = 1, 2, \ldots, T$. Jobs are assumed to arrive at the beginning of the time slot, and are assigned to one of the available workers or discarded immediately. Moreover, we assume that jobs will be completed at the end of time slots. Note that the assumption of jobs being completed at the end of time slots is superfluous, since workers who complete a job in the middle of a slot cannot be assigned to any job until the beginning of the next time slot. A job arrives at the beginning of each time slot with probability p. This job arrival process is defined as *geometric arrivals*, since the inter-arrival times are IID geometric random variables. The geometric arrival process is the discrete-time counterpart for a Poisson process. If a time slot has no job arrival, we assume that there is a virtual job with value zero arriving at the beginning of this slot [4]. Therefore, each job keeps the assigned worker busy over the time period $[a_n, a_n + l_n)$ ($a_n \in \mathbb{Z}^+$ and $l_n \in \mathbb{Z}^+ \bigcup \{0\}$ for all n).

We consider a sequence of jobs that arrives during the time interval $[1, T]$. Define a sequence of assignment variables, $X_{n,m} \in \{0, 1\}$, as an indicator of the n^{th} job assigned to worker w_m (we use the success rate to refer to a worker). Job assignments are assumed to be irrevocable and non-preemptive. Moreover, a job can be assigned to at most one worker, and a worker can be assigned to at most one job at a time. An algorithm $\mathcal{A}$ defines a sequence of assignment variables, $\{X^{\mathcal{A}}_{n,m}\}$ for $n = 1, 2, \ldots, T$ and $m = 1, 2, \ldots, m_0$.

The objective of RSSAP is to maximize the expected *reward* for assigning all the jobs using available workers, where the expectation is with respect to the distribution of job sequences and the randomization of the algorithm

(for randomized algorithm only). The reward for assigning job J_n using algorithm $\mathcal{A}$, denoted by $R_{\mathcal{A}}(J_n)$, is given by the product of the job values and the assigned workers' success rates,

$$R_{\mathcal{A}}(J_n) = \sum_{m=1}^{m_0} X_{n,m}^{\mathcal{A}} v_n w_m.$$

Dynamic programming can be applied to obtain the optimal offline algorithm for a given job sequence. However, dynamic programming is intractable due to the dimension of states [67]. Therefore, we seek approximation algorithms and use *competitive ratio* to evaluate the resulting online algorithms.

Definition 4. *Let OPT denote the optimal (maximal) expected reward for assigning job sequences in RSSAP. Let $R(\mathcal{A})$ denote the reward for assigning such job sequences using algorithm $\mathcal{A}$. Then algorithm $\mathcal{A}$ is said to be $\gamma_{\mathcal{A}}$-competitive where*

$$\gamma_{\mathcal{A}} \triangleq OPT/\mathbb{E}[R(\mathcal{A})],$$

and the expectation is taken with respect to the distribution of job sequences and the randomization of $\mathcal{A}$ (for randomized algorithm only).

5.2 Approximation Algorithms for IID Job Arrivals

This section provides approximation algorithms for RSSAP with IID job arrivals. Two cases are considered: (a) all jobs have the same fixed length and IID values with distribution $F_v(v)$; (b) all jobs have IID memoryless length and IID values with distribution $F_v(v)$. The proposed approximation algorithms for both cases are Greedy SSAP optimal policies.

5.2.1 Preliminary: classic SSAP optimal policy

[8] introduces the sequential stochastic assignment problem (SSAP), where T workers with success rates $\tau_1 \leq \tau_2 \leq \ldots \leq \tau_T$ are assigned to T sequentially arriving jobs with values $\{\mathcal{C}_t\}_{t=1}^T$ (random variables) revealed upon arrival. This problem is referred to as the T-*depth* SSAP problem. The objective is to maximize the total expected reward $\mathbb{E}[\sum_{t=1}^T \tau_{j_t} \mathcal{C}_t]$, where j_t is the index of the worker assigned to the t^{th} job with value $\mathcal{C}_t$. The optimal policy uses

recursive equations to compute threshold values for each job assignment, which motivates our proposed approximation algorithms for RSSAP.

Theorem 25 ([8]). *For the t^{th} job arrival with job value $\mathcal{C}_t$, there are $T-t+1$ workers available, $t = 1, 2, \ldots, T$. The thresholds for $\mathcal{C}_t$ are given by $-\infty = a_0^{T-t+1} \leq a_1^{T-t+1} \leq \ldots \leq a_{T-t+1}^{T-t+1} = +\infty$, where*

$$a_i^{T-t+1} = \int_{a_{i-1}^{T-t}}^{a_i^{T-t}} x dF_{\mathcal{C}}(x) + a_{i-1}^{T-t} F_{\mathcal{C}}(a_{i-1}^{T-t}) + a_i^{T-t}(1 - F_{\mathcal{C}}(a_i^{T-t})),\ i = 1, 2, \ldots, T-t. \tag{5.1}$$

If $\mathcal{C}_t \in (a_{i-1}^{T-t}, a_i^{T-t}]$, then the worker with the i^{th} smallest success rate among the $T-t+1$ available workers is assigned to the t^{th} job under the optimal policy (referred to as the SSAP optimal policy). Moreover, a_i^{T-t} is the expected job value that is assigned to the worker with i^{th} smallest success rate among the $T-t$ available workers for $i = 1, 2, \ldots, T-t$.

One important property of the SSAP optimal policy is related to order statistics of IID job values: a_i^t given by (5.1) is the expectation of the i^{th} smallest job value out of t IID job values, which follows from Hardy's lemma [68].

[4] considers a Generalized SSAP problem (GSSAP), where at each time t a job arrives with some probability $0 < p \leq 1$. Therefore, the total number of arriving jobs is a random variable. They show that this GSSAP is equivalent to an SSAP with cdf given by

$$F_G(x) = (1-p) + pF_{\mathcal{C}}(x). \tag{5.2}$$

That is, no job arrival is treated as an arriving job with value zero. The optimal policy for the GSSAP is given by Theorem 25 with $F_G(x)$ substituted for $F_{\mathcal{C}}(x)$. $F_G(x)$ is referred to the *refined value distribution*.

5.2.2 Equal-length job sequences

This section considers equal-length job sequences. Let $l_0 \in \mathbb{Z}^+ \bigcup \{0\}$ denote the length of each job. If $l_0 \leq 1$, then the optimal solution is to assign every arriving job to the worker with the largest success rate, who is able to complete all jobs. We assume $l_0 \geq 2$ in the following analysis. Sections 5.2.2.1

and 5.2.2.2 consider a single available worker while Section 5.2.2.3 discusses the case of multiple workers.

Several definitions are needed. The time interval starting from the arrival time (included) of an assigned job until its completion time (not included) is referred to as the *blocking window* of the assigned job. Jobs arriving after the assigned job and during this blocking window are *blocked* by this assigned job (the assigned job is in its own blocking window) since assignments are irrevocable and non-preemptive. Therefore, only one job can be assigned within each blocking window. In the case of IID arrivals, all blocking windows have the same distribution, given the same size of l_0.

5.2.2.1 A single worker

Since only one worker is available, we assume that $w_1 = 1$. We divide the time axis into *stages*, which are time intervals of length $2l_0$ and are numbered sequentially. For example, the time interval starting from $t = 1$ and lasting until $t = 2l_0$ (including both ends, i.e., $[1, 2l_0]$) is referred to as *stage one*, the time interval from $t = 2l_0 + 1$ to $t = 4l_0$ is referred to as *stage two*, and so forth. We propose the Greedy Threshold algorithm, which is a threshold algorithm based on the optimal policy for SSAP.

The intuition behind the Greedy Threshold algorithm is two-fold: (1) the worker should be used as many times as possible, and (2) the worker should be assigned to a job with the highest value (in expectation) whenever the worker is available. According to the Greedy Threshold algorithm, the worker is used at least once every $2l_0$ time slots and assigned to the job with the highest value among blocked jobs, in expectation.

To compute the competitive ratio of the Greedy Threshold algorithm, we first derive an upper bound for the optimal expected reward. Then we give a lower bound for the expected reward using the Greedy Threshold algorithm. Note that the Greedy Threshold algorithm is a deterministic algorithm, and hence the expectation is taken over the distribution of the job sequence.

Lemma 5. *An upper bound for the optimal expected reward for assigning equal-length jobs to a single reusable worker, R_E^*, is*

$$R_E^* \leq (\lfloor T/l_0 \rfloor + 1)a_{l_0}^{l_0},$$

Algorithm 4 Greedy Threshold Algorithm

Compute the refined cdf $F_G(v)$ for job values using (5.2).
Compute the threshold values in a l_0-depth SSAP problem with job value distribution $F_G(v)$, $\{a_i^j\}$. Then a_i^j is the expected value of the i^{th} smallest job value among j IID jobs with cdf $F_G(v)$, for $i = 1, 2, \ldots, j$ and $j = 1, 2, \ldots, l_0$.
Beginning at stage one (i.e., from $t = 1$ to $t = 2l_0$).
while $t \leq T$ **do**
Re-index the time slots in each stage as $t' = 1$ to $t' = 2l_0$.
If a job J arrives at time t', then J will be assigned to worker w_1 if and only if

$$v(J) \geq a_{l_0 - t'}^{l_0 - t'}, \tag{5.3}$$

for $t' = 1, 2, \ldots, l_0$ with $a_0^0 = 0$.
If worker w_1 is assigned a job, let t^* denote the arrival time of the job. Then the job will be completed at $t = t^* + l_0$. Therefore, the next stage is defined as starting from $t = t^* + l_0$ until $t = t^* + 3l_0 - 1$. Otherwise, let t^* denote the re-indexed time $t' = l_0$, and the next stage is defined as starting from $t = t^* + 1$ until $t = t^* + 2l_0$.
end while

where T is the arrival time of the last job, l_0 is the job length, and $a_{l_0}^{l_0}$, defined by (5.1)*, is the expectation of the largest job value for l_0 IID job values.*

Proof: The proof uses properties of the IID arrivals of the job sequence. We provide upper bounds for two elements: (1) the total number of completed job assignments; (2) the expected job value for each job assignment. The product of upper bounds for these two elements provides an upper bound for the optimal expected reward.

Consider the total number of job assignments the worker is able to complete. Since each job will take l_0 time slots to complete, the total number of completed assignments is at most $(\lfloor T/l_0 \rfloor + 1)$, which is achieved by assigning jobs to the worker back-to-back (the last assignment may be completed after T).

Consider the expected job value assigned to the worker. Each time a job is assigned to the worker, the job will incur a blocking window of length l_0. Therefore, the expected job value assigned to the worker has an upper bound given by the expectation of the largest job value out of l_0 IID jobs with cdf $F_G(v)$, which is $a_{l_0}^{l_0}$ from Theorem 25. Since job arrivals in the entire job sequence are IID, each blocking window of length l_0 has the same

distribution, and hence each expected job value assigned to the worker has an upper bound given by $a_{l_0}^{l_0}$.

The product of these two upper bounds gives an upper bound for the optimal expected reward. ■

Proposition 4 provides a lower bound for the expected reward using the Greedy Threshold algorithm.

Proposition 4. *A lower bound for the expected reward using the Greedy Threshold algorithm,* $\mathbb{E}[R_{GT}]$*, is*

$$\mathbb{E}[R_{GT}] \geq (\frac{1}{2}\lfloor\frac{T}{l_0}\rfloor - \frac{1}{2l_0})a_{l_0}^{l_0},$$

where the expectation is taken over the distribution of the job sequence.

Proof: Since the Greedy Threshold algorithm treats each stage the same way and job arrivals are IID, the expected reward using the Greedy Threshold algorithm is the product of the number of stages and the expected reward in each stage.

First, we consider the number of stages. From the Greedy Threshold algorithm, the worker is assigned once in each stage. Therefore, the total number of stages is the total number of jobs completed by the worker, denoted by $N(S)$. Note that the actual size of a stage is less than or equal to $2l_0$, and hence $N(S)$ has a lower bound given by

$$N(S) \geq \lfloor\frac{T}{2l_0}\rfloor \geq \frac{T-1}{2l_0} \geq \frac{1}{2}\lfloor\frac{T}{l_0}\rfloor - \frac{1}{2l_0}. \tag{5.4}$$

Next, we consider the expected reward for the Greedy Threshold algorithm in each stage. Since the threshold values for the l_0-depth SSAP problem are used for assigning jobs to the worker, then from Theorem 25, the expected reward for each stage is $a_{l_0}^{l_0}$.

Combining the number of stages and the expected reward in each stage together leads to the desired result. ■

Theorem 26 provides the competitive ratio of the Greedy Threshold algorithm on equal-length job sequences.

Theorem 26. *The Greedy Threshold algorithm is 2-competitive for IID equal-length job sequences.*

Proof: The result follows directly from Lemma 5 and Proposition 4. In particular, let γ_{GT} denote the competitive ratio of the Greedy Threshold algorithm. Then,

$$\gamma_{GT} = \frac{R_E^*}{\mathbb{E}[R_{GT}]} \leq \frac{(\lfloor T/l_0 \rfloor + 1)a_{l_0}^{l_0}}{(\frac{1}{2}\lfloor \frac{T}{l_0} \rfloor - \frac{1}{2l_0})a_{l_0}^{l_0}} \to 2,$$

as $T \to +\infty$. ■

Theorem 26 holds for any job value distribution $F_v(v)(F_G(v))$ with finite mean. Moreover, Theorem 26 holds for jobs with non-integer (continuous) lengths, as long as job arrivals only happen at the beginning of time slots.

A stronger competitive ratio can be obtained for specific distributions. As an example, we consider two continuous distributions: the uniform distribution and the exponential distribution. Since the arrival process is geometric with probability p, we consider the refined job value distribution $F_G(v)$ as a uniform distribution or an exponential distribution and provide the corresponding $F_{\mathcal{C}}(v)$.

If $F_G(v)$ is a uniform distribution, the original value distribution $F_{\mathcal{C}}(v)$ can either be a uniform distribution on the half-open interval $(0, B]$ with a geometric arrival process $p = B/(B+1)$ for $B > 0$ or any uniform distribution with a geometric arrival process $p = 1$. Corollary 3 provides a competitive ratio of 3/2 using the Greedy Threshold algorithm.

Corollary 3. *The Greedy Threshold algorithm is 3/2-competitive for IID equal-length job sequences, when the refined value distribution is a uniform distribution on* $[c, d]$, $c,\ d \in \mathbb{R}^+$.

Proof: We derive a lower bound for the expected reward using the Greedy Threshold algorithm. Let a_i^j denote the i^{th} smallest threshold value for a j-depth SSAP problem, for $i = 1, 2, \ldots, j$ and $j = 1, 2, \ldots, n$, for an n-depth SSAP problem with this uniform distribution. Then $a_i^j = c + \frac{i}{j+1}(d-c)$ from (5.1).

Consider the expected length of each stage using the Greedy Threshold algorithm. Let t^* (re-indexed within the stage) denote the time slot that the

worker gets assigned. Then,

$$\begin{aligned}\mathbb{E}[t^*] =& 1 \times \mathbb{P}(v(J_1) > a_{l_0-1}^{l_0-1}) \\ &+ \sum_{k=2}^{l_0} k\mathbb{P}(v(J_k) > a_{l_0-k}^{l_0-k}, v(J_{k-1}) \leq a_{l_0-k+1}^{l_0-k+1}, \ldots, v(J_1) \leq a_{l_0-1}^{l_0-1}) \\ =& 1\frac{1}{l_0+1} + 2\frac{l_0}{l_0+1}\frac{1}{l_0} + \ldots + l_0\frac{1}{l_0+1} \\ =& \frac{l_0}{2},\end{aligned}$$

where $v(J_k)$ is the job value that arrives at the k^{th} (re-indexed) time slot in the stage. Note that after each assignment, the worker needs l_0 slots to complete the assigned job. Therefore, the expected length of each stage is $l_0/2 - 1 + l_0 = 3l_0/2 - 1 \leq 3l_0/2$.

Let L_i^U denote the length of stage i for $i = 1, 2, \ldots, N_L$, where N_L^U (a random variable) denotes the total number of stages that have occurred by time T (the last stage may be completed after T). Then $\{L_i^U\}$ are IID random variables with mean $3l_0/2-1$. Since the number of stages that have occurred by any time t is a renewal process, N_L^U is a stopping rule for $\{L_i^U\}$. By the definition of N_L^U and Wald's identity,

$$\mathbb{E}[\sum_{i=1}^{N_L^U} L_i] = \mathbb{E}[L_i^U]\mathbb{E}[N_L^U] \geq T \qquad \Rightarrow \mathbb{E}[N_L^U] \geq T/\mathbb{E}[L_i^U] = \frac{2T}{3l_0}.$$

Let R_i^U denote the reward for stage i. Then from Theorem 25, $\{R_i\}$ are IID random variables with expectation $a_{l_0}^{l_0}$. The expected reward for assigning all jobs using the Greedy Threshold algorithm, denoted by R_{GT}^U, has a lower bound given by

$$\mathbb{E}[R_{GT}^U] = \mathbb{E}[\sum_{i=1}^{N_L^U} R_i^U] = \mathbb{E}[N_L^U]\mathbb{E}[R_i^U] = \mathbb{E}[N_L^U]a_{l_0}^{l_0} \geq \frac{2T}{3l_0}a_{l_0}^{l_0},$$

where the second equality follows from Wald's identity. From the upper bound for the optimal expected reward given by Lemma 5, the competitive ratio of the Greedy Threshold algorithm is

$$\gamma_{GT}^U = \frac{R_E^*}{\mathbb{E}[R_{GT}^U]} \leq \frac{(\lfloor T/l_0 \rfloor + 1)a_{l_0}^{l_0}}{\frac{2T}{3l_0}a_{l_0}^{l_0}} \to 3/2,$$

as $T \to +\infty$. ■

If $F_G(v)$ is an exponential distribution, then the original distribution $F_{\mathbb{e}}(v)$ and the refined distribution $F_G(v)$ have the following relationship:

$$\begin{aligned} F_G(v) &= 1 - p + pF_{\mathbb{e}}(v) = 1 - p + p(1 - \exp(-\lambda v)) \\ &= 1 - p\exp(\lambda v) = 1 - \exp\left(-\lambda(v - \frac{\ln p}{\lambda})\right), \end{aligned}$$

where $F_{\mathbb{e}}(v)$ is assumed to be an exponential distribution with parameter λ and $F_G(v)$ is an exponential distribution with parameter λ and a shift of $\ln p/\lambda$ to the left with respect to v. We provide a method to compute a tighter competitive ratio for exponentially distributed job values as follows.

Consider the expected length of each stage. Let L_i^E denote the length of stage i under the Greedy Threshold algorithm for equal-length jobs with exponential value distribution, $i = 1, 2, \ldots, N_L^E$, where N_L^E denotes the total number of stages that have occurred by time T (the last stage may be completed after T). Then $\{L_i^E\}$ are IID random variables by definition. We use the following theorem for order statistics to compute the threshold values for the l_0-depth SSAP problem:

Theorem 27 (*Theorem 6.5*, [69])**.** *Let $Y_1, Y_2, \ldots, Y_n$ be n IID exponential random variables with parameter λ. Let $Y_{(1)} \le Y_{(2)} \le \ldots \le Y_{(n)}$ denote their order statistics. Define $W_0 \triangleq Y_{(1)}$ and $W_i \triangleq Y_{(i+1)} - Y_{(i)}$, $i = 1, 2, \ldots, n-1$. Then $\{W_i\}$ are independent and exponentially distributed with parameter $\lambda(n-i)$, $i = 0, 1, \ldots, n-1$. Moreover,*

$$\mathbb{E}[Y_{(n)}] = \mathbb{E}[\sum_{j=0}^{n-1} W_j] = \sum_{j=0}^{n-1} \mathbb{E}[W_j] = \sum_{j=0}^{n-1} \frac{1}{\lambda(n-j)} = \frac{H_n}{\lambda},$$

where H_n is the n^{th} harmonic number.

For the l_0-depth SSAP problem with exponential job value distribution, from Theorem 27, the threshold values in each stage are

$$a_i^i = \frac{H_i}{\lambda} + \frac{\ln p}{\lambda}, \tag{5.5}$$

for $i = 1, 2 \ldots, l_0 - 1$. Therefore,

$$\begin{aligned}
\mathbb{P}_1 &\triangleq \mathbb{P}(v(J_1) > a_{l_0-1}^{l_0-1}) = \exp(-H_{l_0-1}), \\
\mathbb{P}_i &\triangleq \mathbb{P}(v(J) > a_{l_0-i}^{l_0-i}, v(J_1) \leq a_{l_0-1}^{l_0-1}, \ldots, v(J_{i-1}) \leq a_{l_0-i+1}^{l_0-i+1}) \\
&= \left(\Pi_{j=l_0-i+1}^{l_0-1}(1 - \exp(-H_j))\right) \exp(-H_{l_0-i}), \qquad \text{for } i = 2, 3, \ldots, l_0. \quad (5.6)
\end{aligned}$$

Then the expected length of each stage is

$$\mathbb{E}[L_i^E] = l_0 + \sum_{j=0}^{l_0-1} jP_{j+1}.$$

Let R_{GT}^E denote the reward for assigning equal-length jobs with exponentially distributed values. Note that the expected reward for each stage is $a_{l_0}^{l_0}$. Then from Wald's identity,

$$\mathbb{E}[R_{GT}^E] = \mathbb{E}[N_L^E] a_{l_0}^{l_0} \geq \frac{T}{\mathbb{E}[L_i^E]} a_{l_0}^{l_0} = \frac{T}{l_0 + \sum_{j=0}^{l_0-1} jP_{j+1}} \left(\frac{H_{l_0}}{\lambda} + \frac{\ln p}{\lambda}\right), \quad (5.7)$$

where the inequality follows since the last stage may be completed after T. Let γ_{GT}^e denote the competitive ratio of the Greedy Threshold algorithm for equal-length jobs with exponentially distributed values. Combining the lower bound provided in (5.7) and the upper bound for the optimal expected reward provided in Lemma 5 leads to

$$\gamma_{GT}^E = \frac{R_E^*}{\mathbb{E}[R_{GT}^E]} \leq \frac{(\lfloor T/l_0 \rfloor + 1) a_{l_0}^{l_0}}{T/(l_0 + \sum_{j=0}^{l_0-1} jP_{j+1}) a_{l_0}^{l_0}} \to 1 + \frac{\sum_{j=0}^{l_0-1} jP_{j+1}}{l_0}. \quad (5.8)$$

Note that $(\sum_{j=0}^{l_0-1} jP_{j+1})/l_0 \leq 1$ (from the definition for SSAP optimal policy), and hence the competitive ratio given by (5.8) is less than or equal to 2, which is a stronger result than Theorem 26.

5.2.2.2 Fixed-threshold algorithm for a single worker

This section considers a class of Greedy algorithms using a single fixed threshold, as a comparison with the Greedy Threshold algorithm proposed in Section 5.2.2.1. For general distributions of job values, the competitive ratio of the Greedy algorithm with a fixed threshold (referred to as the Fixed-

threshold algorithm) does not have a simple closed-form expression. However, if the job value follows a uniform distribution, the expression for the competitive ratio of the Fixed-threshold algorithm can be obtained.

Let $\hat{v}$ denote the threshold for the job value under the Fixed-threshold algorithm: if $v(J) \geq \hat{v}$ and the worker is available, assign job J to the worker; otherwise, discard the job. We will determine the value of $\hat{v}$ later. To analyze the Fixed-threshold algorithm, divide the time axis into stages, each spanning the interval between the time slot when the worker is first available after completing a previous job and the time slot when the worker completes one job assignment. For example, the first stage extends from time $t = 1$ until the time, denoted by t_1, when the worker completes the first job assignment. Then the second stage extends from $t_1 + 1$ until the time when the worker completes the second job assignment. That is, one job is completed in each stage.

Let L_i denote the length of stage i, for $i = 1, 2, \ldots, N_S$, where N_S denotes the total number of stages that have occurred by time T (the last stage may be completed after T). Let R_i denote the reward achieved by the Fixed-threshold algorithm in stage i. Then, $\{L_i\}$ and $\{R_i\}$ are both IID random variables, since the Fixed-threshold algorithm uses the same threshold value for all job assignments. Moreover, each stage consists of two parts: (1) the worker waits to be assigned in the first part, whose length is a geometrically distributed random variable with parameter p_L given by

$$\begin{aligned} p_L &= \mathbb{P}(\text{the worker is assigned at a time slot } t) \\ &= \mathbb{P}\left(\text{job J is assigned to the worker} \mid \text{job J arrives}\right) \mathbb{P}\left(\text{job J arrives}\right) \\ &= p\mathbb{P}(v(J) \geq \hat{v}); \end{aligned}$$

and (2) the worker completes the assigned job in the second part, whose length is a constant l_0. Therefore,

$$\mathbb{E}[L_i] = \frac{1}{p\mathbb{P}(v(J) \geq \hat{v})} + l_0 - 1, \tag{5.9}$$

where the minus one is because the geometric distribution of the first part starts from zero. The expectation of R_i is given by $\mathbb{E}[R_i] = \mathbb{E}[v(J) \mid v(J) \geq \hat{v}]$, where the expectation is taken with respect to the original value distribution $F_v(v)$.

Since the number of stages that have occurred by any time t is a renewal process, N_S is a stopping rule for both $\{L_i\}$ and $\{R_i\}$. Let R_{FT} denote the reward using the Fixed-threshold algorithm for equal-length jobs. By Wald's identity,

$$\mathbb{E}[\sum_{i=1}^{N_S} L_i] = \mathbb{E}[N_S]\mathbb{E}[L_i] \geq T \Rightarrow \mathbb{E}[N_S] \geq \frac{T}{\mathbb{E}[L_i]},$$
$$\mathbb{E}[R_{FT}] = \mathbb{E}[\sum_{i=1}^{N_S} R_i] = \mathbb{E}[N_S]\mathbb{E}[R_i] \geq T\frac{\mathbb{E}[r_i]}{\mathbb{E}[L_i]}. \tag{5.10}$$

Therefore, maximizing the lower bound for $\mathbb{E}[R_{FT}]$ is the same as maximizing the ratio $\mathbb{E}[r_i]/\mathbb{E}[L_i]$. Substituting (5.9) into (5.10) leads to the following optimization problem:

$$\max_{\hat{v}\in[v_{min},v_{max}]} \frac{\mathbb{E}[v(J) \mid v(J) \geq \hat{v}]}{\frac{1}{p\mathbb{P}(v(J)\geq\hat{v})} + l_0 - 1}, \tag{5.11}$$

where $[v_{min}, v_{max}]$ is the support for the job value. To obtain the optimal threshold $\hat{v}$, take the derivative of (5.11) with respect to $\hat{v}$ and set the derivative to zero,

$$\frac{\hat{v}}{(\hat{v} - \mathbb{E}[v|v \geq \hat{v}])\mathbb{P}(v \geq \hat{v})} = p(l_0 - 1). \tag{5.12}$$

In general, there is no closed-form solution to (5.12), and hence the specific choice of a threshold for job values $\hat{v}$ depends on parameters of the distribution. We take uniform distributions and exponential distributions as examples.

If the job value is uniformly distributed on $[A, B]$ (i.e., $F_v(v) = (v - A)/(B - A)$ for $v \in [A, B]$), then for $A \leq \hat{v} \leq B$, (5.11) becomes

$$\max_{A\leq\hat{v}\leq B} \frac{(\hat{v} + B)/2}{(B - A)/(p(B - \hat{v})) + l_0 - 1} = \max_{A\leq\hat{v}\leq B} \frac{p}{2}\frac{B^2 - \hat{v}^2}{B - A + (l_0 - 1)p(B - \hat{v})}. \tag{5.13}$$

To obtain the solution to (5.13), change the variable by $x = B - \hat{v}$. Then $2B - x = B + \hat{v}$ and $0 \leq x \leq B - A$. Substituting x for $\hat{v}$ and taking the

derivative of (5.13) with respect to x lead to

$$\begin{aligned}&\frac{(2B-2x)(B-A+(l_0-1)px)-(2B-x)x(l_0-1)p}{(B-A+(l_0-1)px)^2}\\=&\frac{-(l_0-1)p(x^2+2\frac{B-A}{(l_0-1)p}x)+2B(B-A)}{(B-A+(l_0-1)px)^2}.\end{aligned}\tag{5.14}$$

Then the optimal value for x is given by setting (5.14) to zero as $g(x) \triangleq -(l_0-1)p(x^2+2\frac{B-A}{(l_0-1)p}x)+2B(B-A)=0$. Let x^* denote the optimal value for x, which depends on the value of A, B, l_0 and p. If $2B/(B-A) \geq (l_0-1)p+2$, then $g(x) \geq 0$ for all $0 \leq x \leq B-A$, and hence (5.13) is a monotonically increasing function of x and a monotonically decreasing function of $\hat{v}$. Therefore, the maximum of (5.13) is achieved by setting $\hat{v}=A$. Let R_{FT}^U denote the reward using the Fixed-threshold algorithm for equal-length jobs with values following a uniform distribution. A lower bound for the expected reward using the threshold $\hat{v}=A$ is given by substituting A for $\hat{v}$ in (5.13), which gives

$$\mathbb{E}[R_{FT}^U] \geq T\frac{p}{2}\frac{B+A}{1+(l_0-1)p}.\tag{5.15}$$

From Lemma 5, the optimal expected reward for jobs with values uniformly distributed on $[A,B]$ has an upper bound given by

$$R_E^* \leq \lfloor T/l_0 \rfloor (B-\frac{B-A}{l_0+1}) \leq \frac{T}{l_0}\frac{l_0B+A}{l_0+1}.\tag{5.16}$$

Let γ_{FT}^U denote the competitive ratio of the Fixed-threshold algorithm. Then combining (5.15) and (5.16) leads to

$$\gamma_{FT}^U = \frac{R_E^*}{\mathbb{E}[R_{FT}]} \leq \frac{l_0B+A}{l_0(l_0+1)}\frac{2(1+(l_0-1)p)}{p(B+A)} \leq 2\frac{1+(l_0-1)p}{(l_0+1)p}.$$

Note that $1+(l_0-1)p \leq (l_0+1)p$ for $p \geq 1/2$. Therefore, the competitive ratio of the Fixed-threshold algorithm is less than 2 for $p \geq 1/2$.

If the job value is exponentially distributed with parameter λ (i.e., $F_v(v) = 1-\exp(-\lambda v)$ for $v \geq 0$), then (5.11) becomes

$$\max_{\hat{v}\geq 0} p\frac{\int_{\hat{v}}^{+\infty} v\lambda\exp(-\lambda v)dv}{1+(l_0-1)p\exp(-\lambda\hat{v})} = \max_{\hat{v}\geq 0}\frac{p}{\lambda}\frac{\lambda\hat{v}+1}{(l_0-1)p+\exp(\lambda\hat{v})}.\tag{5.17}$$

Taking the derivative of (5.17) with respect to $\hat{v}$ and setting the derivative to zero lead to

$$\lambda \hat{v} \exp(\lambda \hat{v}) = (l_0 - 1)p. \tag{5.18}$$

Let $\hat{v}^*$ denote the solution to (5.18). Then $\lambda \hat{v}^* < (l_0 - 1)p$. Let R_{FT}^E denote the reward using the Fixed-threshold algorithm for equal-length jobs with values following an exponential distribution. Substituting the value of $\exp(\lambda \hat{v}^*)$ from (5.18) back into (5.17) leads to

$$\mathbb{E}[R_{FT}^E] \geq T \frac{\hat{v}^*}{l_0 - 1}.$$

From Lemma 5 and (5.5), an upper bound for the optimal expected reward for jobs with values following an exponential distribution is

$$R_E^* \leq \lfloor T/l_0 \rfloor (\frac{l_0(l_0+1)}{2\lambda} + \frac{\ln p}{\lambda}) \leq T(\frac{l_0+1}{2\lambda} + \frac{\ln p}{l_0 \lambda}).$$

Let γ_{FT}^E denote the competitive ratio of the Fixed-threshold algorithm for jobs with values following an exponential distribution. Then,

$$\gamma_{FT}^E = \frac{R_E^*}{\mathbb{E}[R_{FT}^E]} \leq (\frac{l_0+1}{2\lambda} + \frac{\ln p}{l_0 \lambda}) \frac{l_0 - 1}{\hat{v}^*}. \tag{5.19}$$

5.2.2.3 Multiple workers

This section extends the results in Section 5.2.2.1 to the case of multiple workers. Suppose that there are m_0 workers available, with success rates $w_1 \leq w_2 \leq \ldots \leq w_{m_0}$. Similar to the case of a single worker, the time axis is divided into stages of length $2l_0$.

If $m_0 < l_0$, these workers may not be able to complete all jobs in a sequence. On the other hand, if $m_0 \geq l_0$, there will always be redundant workers, and workers with the m_0 largest success rates are able to complete all the jobs in a job sequence. Lemma 6 gives an upper bound for the optimal expected reward using multiple workers.

Lemma 6. *An upper bound for the optimal expected reward for assigning*

equal-length job sequences to multiple workers, denoted by $R^*_{m_0}$, is

$$R^*_{m_0} \leq (\lfloor T/l_0 \rfloor + 1) \sum_{i=l_0-\min\{m_0,l_0\}+1}^{l_0} a_i^{l_0} w_{m_0-l_0+i}, \tag{5.20}$$

where m_0 is the number of workers with success rates $w_1 \leq w_2 \leq \ldots \leq w_{m_0}$, T is the arrival time of the last job, l_0 is the common job length, and $\{a_i^{l_0}\}$, defined by (5.1)*, are the expectations of the order statistics of l_0 IID job values with cdf $F_G(v)$, for $i = 1, 2, \ldots, l_0$.*

Proof: See Appendix B. ■

The algorithm for RSSAP with multiple workers, the *Greedy SSAP-stage algorithm*, is similar to the Greedy Threshold algorithm. The time axis is divided into $2l_0$-length stages, and each worker (if $m_0 > l_0$, then only workers with the l_0 largest success rates are used) is assigned to one job in each stage. At the beginning of each stage, all workers complete previously assigned jobs and become available. The only difference is that workers have their own threshold values, which is a result of the heterogeneity of workers and Hardy's lemma [68]. The l_0-depth SSAP optimal policy is applied in the first half of each stage: the worker with the largest success rate is assigned to the job with the largest value (in expectation) out of l_0 IID jobs (i.e., jobs in a blocking window); the worker with the second largest success rate is assigned to the job with the second largest value (in expectation) out of l_0 IID jobs; and so on. For simplicity, if $l_0 > m_0$, add $l_0 - m_0$ virtual workers with success rates zero and refer the virtual and original workers together as workers with success rates $w'_1 \leq w'_2 \leq \ldots w'_{l_0}$. Otherwise, if $l_0 \leq m_0$, only use workers with the l_0 largest success rates and refer to them as workers with success rates $w'_1 \leq w'_2 \leq \ldots \leq w'_{l_0}$.

Proposition 5. *A lower bound for the expected reward using the Greedy SSAP-stage algorithm, denoted by $\mathbb{E}[R_{GSS}]$, is*

$$\mathbb{E}[R_{GSS}] \geq 1/2\lfloor T/l_0 \rfloor \sum_{i=l_0-\min\{l_0,m_0\}+1}^{l_0} a_i^{l_0} w_{m_0-l_0+i}, \tag{5.22}$$

where the expectation is taken over the distribution of the job sequence.

Proof: See Appendix B. ■

Algorithm 5 Greedy SSAP-stage Algorithm

Compute the refined cdf $F_G(v)$ for job values using (5.2).
Compute the threshold values in a l_0-depth SSAP problem with job value distribution $F_G(v)$, $\{a_i^j\}$. Then a_i^j is the expected value of the i^{th} smallest job value among j IID jobs with cdf $F_G(v)$, for $i = 1, 2, \ldots, j$ and $j = 1, 2, \ldots, l_0$.
Beginning at stage one (i.e., from $t = 1$ to $t = 2l_0$).
while $t \leq T$ **do**
Reindex the time slots in each stage as $t' = 1$ to $t' = 2l_0$.
At time t', there are workers $w_1^{t'} \leq w_2^{t'} \leq \ldots \leq w_{l_0-t'+1}^{t'}$ available. If a job J arrives, then J is assigned to worker $w_j^{t'}$ if and only if

$$a_{j-1}^{l_0-t'} \leq v(J) < a_j^{l_0-t'}, \tag{5.21}$$

for $t' = 1, \ldots, l_0$ with $a_0^k = 0$ for $k = 1, 2, \ldots, l_0 - 1$.
At time $t_c = 2l_0 + 1$, all workers are available again. Therefore, the next stage is defined as starting from $t = t_c$ till $t = t_c + 2l_0$.
end while

Theorem 28. *The Greedy SSAP-stage algorithm is 2-competitive for IID equal-length job sequences using multiple workers.*

Proof: The result follows directly from Lemma 6 and Proposition 5. Let γ_{GSS} denote the competitive ratio of the Greedy SSAP-stage algorithm. Then,

$$\gamma_{GSS} = \frac{R_M^*}{\mathbb{E}[R_{GSS}]} \leq \frac{(\lfloor T/l_0 \rfloor + 1) \sum_{i=l_0-\min\{m_0,l_0\}+1}^{l_0} a_i^{l_0} w_{m_0-l_0+i}}{1/2 \lfloor T/l_0 \rfloor \sum_{i=l_0-\min\{l_0,m_0\}+1}^{l_0} a_i^{l_0} w_{m_0-l_0+i}} \to 2,$$

as $T \to +\infty$. ■

5.2.3 Memoryless-length job sequences

This section considers memoryless-length job sequences. Values of memoryless-length jobs are independent of their lengths and follow a distribution $F_v(v)$. Lengths of memoryless-length jobs follow a geometric distribution with parameter q (i.e., the probability of a worker completing an assigned job in the time slot). Assigning memoryless-length jobs to workers is the discrete-time counterpart for assigning jobs to servers with exponentially distributed service time in a continuous-time queuing system. The objective is to maximize

the total reward for completed jobs.

Consider m_0 workers with success rates $w_1 \leq w_2 \leq \ldots w_{m_0}$. We apply the Greedy SSAP-stage algorithm for memoryless-length jobs as follows: set the length of each stage as $2\lceil 1/q \rceil$, and use the threshold values computed from an $\lceil 1/q \rceil$-depth SSAP problem in each stage. Note that the expected job length is $\mathbb{E}[l] = \lceil 1/q \rceil$ (the ceiling is due to discrete-time arrivals), and hence the expected size of the blocking window of any assigned job is $\lceil 1/q \rceil$.

To analyze the Greedy SSAP-stage algorithm for memoryless-length jobs, Proposition 6 provides an upper bound for the optimal expected reward and Proposition 7 provides a lower bound for the expected reward for the Greedy SSAP-stage algorithm. The intuition behind the upper bound for the optimal expected reward for memoryless-length jobs is that the time interval $[1, T]$ can be divided into $\lfloor T/l \rfloor$ back-to-back non-overlapping stages with length l, during which each worker is assigned at most one job. Then the SSAP optimal policy for an l-depth SSAP problem can be applied to maximize the expected reward for each stage. Note that this upper bound is not tight, since workers cannot all be available at the beginning of each stage (due to the different thresholds used by different workers in the SSAP optimal policy from Theorem 25). For each feasible stage length l, there is a corresponding upper bound for the optimal expected reward. We want to pick an appropriate l to compute an upper bound for the maximal obtainable expected reward.

We prove Lemma 7 first, which is required for deriving an upper bound for the optimal expected reward. Lemma 7 shows the concavity of the sum of the expected value of the largest order statistics in two groups.

Lemma 7. *For $l_1,\ l_2 \in \mathbb{Z}$ and $1 \leq l_1 < l_2$, the following inequality holds*

$$a_{l_1}^{l_1} + a_{l_2}^{l_2} \geq a_{l_1-1}^{l_1-1} + a_{l_2+1}^{l_2+1}, \tag{5.23}$$

where a_l^l, defined in (5.1)*, is the expected value of the largest order statistics of l IID job values ($a_0^0 \triangleq 0$).*

Proof: From the definition of a_l^l (see (5.1)), the following recursive equations

hold:

$$a_{l_2+1}^{l_2+1} = a_{l_2}^{l_2}\mathbb{P}(v \le a_{l_2}^{l_2}) + \mathbb{E}[v \mid v > a_{l_2}^{l_2}]\mathbb{P}(v > a_{l_2}^{l_2}),$$
$$a_{l_1}^{l_1} = a_{l_1-1}^{l_1-1}\mathbb{P}(v \le a_{l_1-1}^{l_1-1}) + \mathbb{E}[v \mid v > a_{l_1-1}^{l_1-1}]\mathbb{P}(v > a_{l_1-1}^{l_1-1}).$$

Let $\mathbb{P}dx \triangleq dF_v(x)$ denote the differential probability. Therefore,

$$\begin{aligned} a_{l_2+1}^{l_2+1} - a_{l_2}^{l_2} &= \int_{a_{l_2}^{l_2}}^{+\infty} (x - a_{l_2}^{l_2})\mathbb{P}dx, \qquad (5.24)\\ a_{l_1}^{l_1} - a_{l_1-1}^{l_1-1} &= \int_{a_{l_1-1}^{l_1-1}}^{+\infty} (x - a_{l_1-1}^{l_1-1})\mathbb{P}dx \\ &= \int_{a_{l_2}^{l_2}}^{+\infty} (x - a_{l_1-1}^{l_1-1})\mathbb{P}dx + \int_{a_{l_1-1}^{l_1-1}}^{a_{l_2}^{l_2}} (x - a_{l_1-1}^{l_1-1})\mathbb{P}dx \\ &\ge \int_{a_{l_2}^{l_2}}^{+\infty} (x - a_{l_2}^{l_2})\mathbb{P}dx, \end{aligned}$$

where the inequality follows from $a_{l_2}^{l_2} \ge a_{l_1}^{l_1} \ge a_{l_1-1}^{l_1-1}$, since the expected value of the largest order statistics increases with respect to the total number of samples. Therefore,

$$a_{l_1}^{l_1} - a_{l_1-1}^{l_1-1} \ge a_{l_2+1}^{l_2+1} - a_{l_2}^{l_2},$$

which completes the proof. ■

Lemma 7 indicates that when the total number of IID samples and the number of divided groups are fixed, the sum of the expected value of the largest order statistics in each group is maximized by dividing samples into equal-size groups. Corollary 4 indicates that when the numbers of IID samples and equal-size divided groups are fixed, smaller sizes of the divided groups result in a larger sum of the expected value of the largest order statistics in each group.

Corollary 4. *For $h_1, h_2 \in \mathbb{Z}^+$ and $h_1 > h_2$,*

$$h_1 a_{\lfloor T/h_1 \rfloor}^{\lfloor T/h_1 \rfloor} \ge h_2 a_{\lfloor T/h_2 \rfloor}^{\lfloor T/h_2 \rfloor},$$

where a_h^h, defined in (5.1)*, is the expected value of the largest order statistics of h IID job values.*

Proof: We first prove $h_1 a_{\lfloor T/h_1 \rfloor}^{\lfloor T/h_1 \rfloor} \ge (h_1 - 1) a_{\lfloor T/(h_1-1) \rfloor}^{\lfloor T/(h_1-1) \rfloor}$ for any $h_1 > 1$. Con-

struct h_1 groups of IID job values (samples) as follows: $h_1 - 2$ groups with $\lfloor T/(h_1-1) \rfloor$ job values, one group with $(\lfloor T/(h_1-1) \rfloor - 1)$ job values, and one group with one job value. Therefore, from Lemma 7,

$$\begin{aligned} h_1 a_{\lfloor T/h_1 \rfloor}^{\lfloor T/h_1 \rfloor} &\geq (h_1-2) a_{\lfloor T/(h_1-1) \rfloor}^{\lfloor T/(h_1-1) \rfloor} + a_{\lfloor T/(h_1-1) \rfloor - 1}^{\lfloor T/(h_1-1) \rfloor - 1} + a_1^1 \\ &\geq (h_1-2) a_{\lfloor T/(h_1-1) \rfloor}^{\lfloor T/(h_1-1) \rfloor} + a_{\lfloor T/(h_1-1) \rfloor}^{\lfloor T/(h_1-1) \rfloor} \\ &= (h_1-1) a_{\lfloor T/(h_1-1) \rfloor}^{\lfloor T/(h_1-1) \rfloor}, \end{aligned}$$

where (1) the first inequality is an application of Lemma 7: when the numbers of samples and groups are fixed, the sum of the expected value of the largest order statistics in each group is maximized by dividing the samples into equal-size groups; (2) the second inequality follows from Lemma 7 by taking $l_1 = 1$ and $l_2 = \lfloor T/(h_1-1) \rfloor - 1$.

Since $h_1, h_2 \in \mathbb{Z}^+$ and $h_1 > h_2$, then from $h_1 a_{\lfloor T/h_1 \rfloor}^{\lfloor T/h_1 \rfloor} \geq (h_1-1) a_{\lfloor T/(h_1-1) \rfloor}^{\lfloor T/(h_1-1) \rfloor}$ for any $h_1 > 1$,

$$h_1 a_{\lfloor T/h_1 \rfloor}^{\lfloor T/h_1 \rfloor} \geq (h_1-1) a_{\lfloor T/(h_1-1) \rfloor}^{\lfloor T/(h_1-1) \rfloor} \geq \ldots \geq h_2 a_{\lfloor T/h_2 \rfloor}^{\lfloor T/h_2 \rfloor}.$$

■

Lemma 8 generalizes Lemma 7 to the sum of a fixed number of the largest order statistics in each group. Therefore, when the numbers of IID samples and divided groups are fixed, the sum of the expected value of a fixed number of the largest order statistics in each group is maximized by dividing the samples into equal-size groups.

Lemma 8. *For l, l_1 and $l_2 \in \mathbb{Z}^+$ with $1 \leq l < l_1 < l_2$,*

$$\sum_{i=l_1-l+1}^{l_1} a_i^{l_1} + \sum_{i=l_2-l+1}^{l_2} a_i^{l_2} \geq \sum_{i=l_1-l}^{l_1-1} a_i^{l_1-1} + \sum_{i=l_2-l+2}^{l_2+1} a_i^{l_2+1}, \tag{5.25}$$

where $\{a_i^j\}$, defined in (5.1)*, is the expected value of the order statistics of j IID job values.*

Proof: See Appendix B. ■

Corollary 5 is a direct application of Lemma 8 and generalizes Corollary 4 to the sum of a fixed number of the largest order statistics in each group. Therefore, when the numbers of IID samples and divided groups are fixed, smaller sizes of the divided groups result in a larger sum of the expected

value of a fixed number of the largest order statistics in each group. The proof follows similar arguments as that of Corollary 4.

Corollary 5. *For $h_1, h_2 \in \mathbb{Z}^+$, $h_1 > h_2$ and $l \leq T/(2h_1)$,*

$$h_1 \sum_{i=\lfloor T/h_1 \rfloor - l + 1}^{\lfloor T/h_1 \rfloor} a_i^{\lfloor T/h_1 \rfloor} \geq h_2 \sum_{i=\lfloor T/h_2 \rfloor - l + 1}^{\lfloor T/h_2 \rfloor} a_i^{\lfloor T/h_2 \rfloor},$$

where $\{a_i^h\}$, defined in (5.1)*, is the expected value of the order statistics of h IID job values.*

Proof: See Appendix B. ■

Proposition 6 provides an upper bound for the optimal expected reward for assigning memoryless-length jobs to multiple workers.

Proposition 6. *An upper bound for the optimal expected reward for assigning memoryless-length job sequences to multiple workers, denoted by R^*_{MLL}, is*

$$R^*_{MLL} \leq (T/\lceil 1/q \rceil + 1) \sum_{i=\lceil 1/q \rceil - \min\{m_0, \lceil 1/q \rceil\} + 1}^{\lceil 1/q \rceil} a_i^{\lceil 1/q \rceil} w_{m_0 - \lceil 1/q \rceil + i}, \qquad (5.26)$$

where $\{a_i^{\lceil 1/q \rceil}\}$, $i = 1, 2, \ldots, \lceil 1/q \rceil$, defined by (5.1)*, is the expected value of the order statistics of $\lceil 1/q \rceil$ IID job values.*

Proof: We provide upper bounds for the total number of jobs completed by each worker and the expected reward for each completed job.

Let $L_{k,m}$ and $V_{k,m}$ denote the length and value of the k^{th} job assigned to worker w_m, $k = 1, 2, \ldots, N_m$ and $m = 1, 2, \ldots, m_0$, where N_m is the total number of jobs completed by worker w_m. Since jobs have IID memoryless lengths independent of values, then $\{L_{k,m}\}$ are IID random variables with expectation $\lceil 1/q \rceil$. For any fixed m, using Wald's identity,

$$\sum_{k=1}^{N_m} L_{k,m} \leq T + L_{k,N_m} \qquad \Rightarrow \mathbb{E}[\sum_{k=1}^{N_m} L_{k,m}] \leq T + \lceil 1/q \rceil$$
$$\Rightarrow \mathbb{E}[N_m] \lceil 1/q \rceil \leq T + \lceil 1/q \rceil,$$

and hence $\mathbb{E}[N_m] \leq T/\lceil 1/q \rceil + 1$ for $m = 1, 2, \ldots, m_0$.

Consider the expected reward for each job completed by the worker with the largest success rate w_{m_0}. Since the k^{th} assigned job has a blocking window of size L_{k,m_0} (including the job itself), an upper bound for the expected value of each assigned job is the expected value of the largest order statistics out of L_{k,m_0} IID job values. That is, $\mathbb{E}[V_{k,m_0}] \leq a_{L_{k,m_0}}^{L_{k,m_0}}$, with $a_{L_{k,m_0}}^{L_{k,m_0}}$ defined by Theorem 25 (5.1). Therefore, an upper bound for the optimal expected reward for worker w_{m_0} is $\sum_{k=1}^{N_M} a_{L_{k,m_0}}^{L_{k,m_0}}$. Note that $\{L_{k,m_0}\}$ are IID random variables and $\sum_{k=1}^{N_M-1} L_{k,m_0} \leq T$. From Lemma 7, an upper bound $\sum_{k=1}^{N_M} a_{L_0}^{L_0}$ for $\sum_{k=1}^{N_M} a_{L_{k,m_0}}^{L_{k,m_0}}$ is obtained when $L_{k,m_0} = L_0$ for all $k = 1, 2, \ldots, N_M$ (i.e., equal-size divided groups for a fixed number of IID samples). Since $\mathbb{E}[L_{k,m_0}] = \lceil 1/q \rceil$, then the feasible region for L_0 is $L_0 \geq \lceil 1/q \rceil$. Then from Corollary 4, the largest upper bound $\sum_{k=1}^{N_M} a_{L_0}^{L_0}$ is obtained by setting $L_0 = \lceil 1/q \rceil$. Therefore,

$$\mathbb{E}[\sum_{k=1}^{N_M} a_{L_{k,m_0}}^{L_{k,m_0}}] \leq \mathbb{E}[\sum_{k=1}^{N_M} a_{\lceil 1/q \rceil}^{\lceil 1/q \rceil}] \leq \mathbb{E}[N_M] a_{\lceil 1/q \rceil}^{\lceil 1/q \rceil} \leq (T/\lceil 1/q \rceil + 1) a_{\lceil 1/q \rceil}^{\lceil 1/q \rceil}. \quad (5.27)$$

An upper bound for the expected reward using multiple workers is obtained using similar arguments as for using only the worker with the largest success rate. During the execution of the k^{th} job assigned to the worker with the largest success rate w_{m_0}, an upper bound for the expected reward of assigning jobs arriving during this blocking window to workers $w_1, w_2, \ldots, w_{m_0}$ is $\sum_{i=L_{k,m_0}-\min\{m_0,L_{k,m_0}\}+1}^{L_{k,m_0}} a_i^{L_{k,m_0}} w_{m_0-L_{k,m_0}l+i}$ from Theorem 25. Then an upper bound for the expected reward of assigning all jobs to workers is

$$\sum_{k=1}^{N_M} \sum_{i=L_{k,m_0}-\min\{m_0,L_{k,m_0}\}+1}^{L_{k,m_0}} a_i^{L_{k,m_0}} w_{m_0-L_{k,m_0}+i}.$$

From Lemma 8,

$$\sum_{k=1}^{N_M} \sum_{i=L_{k,m_0}-\min\{m_0,L_{k,m_0}\}+1}^{L_{k,m_0}} a_i^{L_{k,m_0}} w_{m_0-L_{k,m_0}+i} \leq \sum_{k=1}^{N_M} \sum_{i=L_0-\min\{m_0,L_0\}+1}^{L_0} a_i^{L_0} w_{m_0-L_0+i}, \quad (5.28)$$

where the upper bound is obtained when $L_{k,m_0} = L_0$ for all $k = 1, 2, \ldots, N_M$ (i.e., equal-size divided groups for a fixed number of IID samples). Since $\mathbb{E}[L_{k,m_0}] = \lceil 1/q \rceil$, then the feasible region for L_0 is $L_0 \geq \lceil 1/q \rceil$. Then from

Corollary 5, the largest upper bound on the right-hand side of (5.28) is obtained by setting $L_0 = \lceil 1/q \rceil$. Therefore, the upper bound for the reward using multiple workers is

$$R^*_{MLL} \leq (T/\lceil 1/q \rceil + 1) \sum_{i=\lceil 1/q \rceil - \min\{m_0, \lceil 1/q \rceil\}+1}^{\lceil 1/q \rceil} a_i^{\lceil 1/q \rceil} w_{m_0 - \lceil 1/q \rceil + i}.$$

■

Proposition 7 provides a lower bound for the expected reward using the Greedy-SSAP stage algorithm for assigning memoryless-length jobs.

Proposition 7. *A lower bound for the expected reward using the Greedy SSAP-stage algorithm for assigning memoryless-length jobs, denoted by* $\mathbb{E}[R_{MLL}]$*, is*

$$\mathbb{E}[R_{MLL}] \geq T/(2\lceil 1/q \rceil) \sum_{i=\lceil 1/q \rceil - \min\{\lceil 1/q \rceil, m_0\}+1}^{\lceil 1/q \rceil} a_i^{\lceil 1/q \rceil} w_{m_0 - \lceil 1/q \rceil + i}, \qquad (5.29)$$

where the expectation is taken over the distribution of the job sequence.

Proof: See Appendix B. ■

Theorem 29 gives the competitive ratio of the Greedy-SSAP stage algorithm.

Theorem 29. *The Greedy SSAP-stage algorithm achieves a competitive ratio of 2 for memoryless-length job sequences using multiple workers.*

Proof: The result follows directly from Propositions 6 and 7. Let γ_{MLL} denote the competitive ratio of the Greedy SSAP-stage algorithm for assigning memoryless-length jobs. Then,

$$\gamma_{MLL} = \frac{R^*_{MLL}}{\mathbb{E}[R_{MLL}]} \leq \frac{(T/\lceil 1/q \rceil + 1) \sum_{i=\lceil 1/q \rceil - \min\{m_0, \lceil 1/q \rceil\}+1}^{\lceil 1/q \rceil} a_i^{\lceil 1/q \rceil} w_{m_0 - \lceil 1/q \rceil + i}}{T/(2\lceil 1/q \rceil) \sum_{i=\lceil 1/q \rceil - \min\{\lceil 1/q \rceil, m_0\}+1}^{\lceil 1/q \rceil} a_i^{\lceil 1/q \rceil} w_{m_0 - \lceil 1/q \rceil + i}} \to 2,$$

as $T \to +\infty$. ■

5.3 Approximation Algorithms for RSSAP with a Random Order of Arrivals

This section assumes that jobs have a random order of arrivals, relaxing the assumption that job values are drawn from a given distribution. Therefore, jobs are randomly ordered such that the i^{th} arriving job is equally likely to have the j^{th} largest value for all $i, j = 1, 2, ..., T$, where T denotes the given total number of jobs. We provide e-competitive and $e^2/(e-1)$-competitive approximation algorithms for equal-length and memoryless-length jobs, respectively. We further extend these results to equal-length and memoryless-length jobs and provide e^p-competitive and $e^{p+1}/(e-1)$-competitive approximation algorithms, respectively, for both geometric with parameter p and a random order of arrivals.

5.3.1 Preliminary: Bipartite online matching

[53] proposes an online algorithm, referred to as BOM, for the weighted bipartite online matching problem. In the weighted bipartite online matching problem, right-side vertices R of an edge-weighted bipartite graph $G = (R \bigcup L, E)$ are given in advance. Left-side vertices arrive one at a time with their edges e and weights of edges $w(e)$ revealed upon each arrival. A matching algorithm decides either to match an arriving left-side vertex to an unmatched right-side vertex or discard the left-side vertex upon each arrival. The objective is to maximize the expected total weight of edges between the matched vertices in a bipartite graph. The left-side vertices are assumed to arrive in a random order. Note that each right(left) vertex can be matched to at most one left(right) vertex.

The BOM algorithm uses the first $\lfloor T/e \rfloor$ arriving vertices for training, where T is the given total number of right-side vertices (also the number of left-side vertices). Beginning from the $(\lfloor T/e \rfloor + 1)^{th}$ arriving vertex, the BOM algorithm matches vertices based on the optimal (offline) matching for the set of vertices observed to date: if the newly arrived left-side vertex is in the optimal matching for the set of all right-side vertices and the set of left-side vertices observed so far and its matched right-side vertex is available, then the BOM algorithm matches the two vertices; otherwise, the BOM algorithm discards the left-side vertex. [53] proves that the BOM algorithm

is e-competitive for the weighted bipartite online matching problem. The BOM algorithm matches each vertex at most once and hence is not directly applicable to RSSAP. However, the BOM algorithm motivates the proposed algorithms for RSSAP with a random order of arrivals and is given in the Appendix B.

5.3.2 Equal-length job sequences

This section considers equal-length job sequences. Let l_0 denote the length of each job and assume $l_0 \geq 2$ as in Section 5.2.2. Let T denote the given total number of jobs. We first consider a special class of geometric arrivals with $p = 1$ (i.e., a job arrives at the beginning of each time slot with probability 1). General geometric arrivals will be discussed in Section 5.3.4. Suppose that there are m_0 available workers, with success rates $w_1 \leq w_2 \leq \ldots \leq w_{m_0}$. We further assume $m_0 = l_0$ (if $m_0 \geq l_0$, we use workers with the l_0 largest success rates; otherwise, $m_0 < l_0$, add $l_0 - m_0$ virtual workers with success rate zero).

We propose the *rolling window algorithm* for scheduling equal-length job sequences with a random order of arrivals. The rolling window algorithm uses a *rolling window* (a time interval) of size l_0, where the optimal weighted matching between workers and jobs is applied: when a job J_t arrives at time t, the algorithm computes the optimal weighted matching between workers $\{w_1, w_2, \ldots, w_{m_0}\}$ and jobs that have arrived during rolling window $W_t = [\max\{1, t - l_0 + 1\}, t]$ (for example, using Hardy's lemma), where weights of edges between workers and jobs are given by products of worker's success rates and job values. If job J_t is assigned to worker w_{m_t} in the optimal matching for W_t, and worker w_{m_t} is available, then assign job J_t to worker w_{m_t}; otherwise, discard job J_t. The intuition behind the rolling window algorithm is two-fold: (1) since every job has a fixed length l_0, then the assignment of a job arriving at time t can only be influenced by jobs that arrive in the rolling window $W_t = [\max\{1, t - l_0 + 1\}, t]$; (2) since jobs follow a random order of arrivals, jobs arriving in each rolling window of size l_0 can be modeled as l_0 jobs uniformly and randomly selected from T jobs, with a uniformly random order among themselves. These two properties are essential in our analysis of the rolling window algorithm.

Algorithm 6 Rolling Window Algorithm

for each time $t \geq l_0$ **do**
Let J_t denote the job arriving at time t.
Update the rolling window $W_t = [\max\{1, t - l_0 + 1\}, t]$.
$S_{W_t} \triangleq$ optimal weighted matching for jobs arriving during W_t and all workers (for example, using Hardy's lemma).
Let w_{m_t} denote the worker assigned to job J_t according to S_{W_t}.
if worker w_{m_t} is available **then** assign job J_t to w_{m_t}.
else discard J_t.
end if
end for

Theorem 30. *The rolling window algorithm is e-competitive for equal-length job sequences with a random order of arrivals.*

Proof:

The proof has the same structure as the proof of e-competitiveness for the BOM Algorithm given by [53]. We prove the e-competitiveness by considering the expected reward for an assignment given by the optimal weighted matching at time t for $t \geq l_0$ and the probability of such an assignment using the rolling window algorithm being feasible.

Let OPT denote the optimal offline reward for scheduling T jobs. Consider the rolling window used at time $t \geq l_0$, denoted by $W_t = [t - l_0 + 1, t]$. Let $OPT(W_t)$ denote the optimal reward for scheduling jobs arriving in the rolling window W_t, assuming all workers are available. Since jobs follow a random arrival order, jobs arriving in W_t, denoted by $\{J_{t-l_0+1}, J_{t-l_0+2}, \ldots, J_t\}$, can be modeled as l_0 jobs uniformly and randomly selected from T jobs. Therefore,

$$\mathbb{E}[OPT(W_t)] \geq \frac{l_0}{T} OPT,$$

where the expectation is taken over the random arrival order of T jobs. Since the arrival order of these l_0 jobs in rolling window W_t is a random permutation of the l_0 jobs, then the expected reward for assignment at any time slot within W_t is equal. That is, $\mathbb{E}[R(t')] = \mathbb{E}[OPT(W_t)]/l_0$, where $R(t')$ denotes the reward at time $t' \in [t - l_0 + 1, t]$. Therefore, the expected reward for the assignment at time t has a lower bound

$$\mathbb{E}[R(t)] \geq \frac{1}{T} OPT, \tag{5.30}$$

where all workers are assumed to be available. However, not all workers may be available at the beginning of a rolling window. We provide a lower bound for the probability that the worker assigned to job J_t under the optimal matching for rolling window W_t, denoted by w_{m_t}, is available at time t.

Since each job has length l_0, assignments before W_t will not influence the availability of worker w_{m_t}. Therefore, worker w_{m_t} is available at time t if no job is assigned to this worker in rolling window W_t. Consider the probability of worker w_{m_t} getting assigned to some job at time $t' \in [t-l_0+1, t]$. According to the rolling window algorithm, this happens if: (1) the assignment of job $J_{t'}$ and worker w_{m_t} is included in the optimal matching for the rolling window $W_{t'} = [t' - l_0 + 1, t']$, and (2) worker w_{m_t} is available at time t'. Since the order of job arrivals in the rolling window $W_{t'}$ follows a random permutation, the probability that the assignment of job $J_{t'}$ and worker w_{m_t} is included in the optimal matching for the rolling window $W_{t'}$ is $1/l_0$. Then

$$\begin{aligned}
&\mathbb{P}(\text{worker } w_{m_t} \text{ is assigned at time } t') \\
=&\mathbb{P}(\text{worker } w_{m_t} \text{ is matched to job } J_{t'})\mathbb{P}(\text{worker } w_{m_t} \text{ is available at } t') \\
\leq&\mathbb{P}(\text{worker } w_{m_t} \text{ is matched to job } J_{t'}) = \frac{1}{l_0}. \qquad (5.31)
\end{aligned}$$

Note that the event that worker w_{m_t} is assigned at time $t' \in [t - l_0 + 1, t - 1]$ is determined by the relative order of job $J_{t'}$ among jobs arriving during $W_{t'} = [t' - l_0 + 1, t']$, which is independent from the relative order of jobs arriving during $[t' - l_0 + 1, t' - 1]$ among themselves. Therefore, the respective events that worker w_{m_t} is assigned at $t' \in [t - l_0 + 1, t - 1]$ are independent. Then

$$\begin{aligned}
&\mathbb{P}(\text{worker } w_{m_t} \text{ is available at } t) \\
=&\mathbb{P}(\text{worker } w_{m_t} \text{ is not assigned in } [t - l_0 + 1, t - 1]) \\
\geq&\Pi_{t'=t-l_0+1}^{t-1}\mathbb{P}(\text{worker } w_{m_t} \text{is not assigned at } t') \qquad (5.32) \\
=&\Pi_{t'=t-l_0+1}^{t-1}\left(1 - \mathbb{P}(\text{worker } w_{m_t} \text{is assigned at } t')\right) \\
\overset{(5.31)}{\geq}&\Pi_{t'=t-l_0+1}^{t-1}\left(1 - \frac{1}{l_0}\right) = (1 - \frac{1}{l_0})^{l_0-1} \geq \frac{1}{e}, \qquad (5.33)
\end{aligned}$$

where inequality (5.32) follows since the respective events that worker w_{m_t} is assigned at $t' \in [t - l_0 + 1, t - 1]$ are independent.

Let R_{RW}^E denote the reward for the rolling window algorithm for equal-length job sequences with a random order of arrivals. Combining (5.30) and (5.33) leads to

$$\begin{aligned}
\mathbb{E}[R_{RW}^E] &= \sum_{t=1}^{T} \mathbb{E}[R(t)]\mathbb{P}(\text{worker } w_{m_t} \text{ is available at } t) \\
&\geq \sum_{t=l_0}^{T} \mathbb{E}[R(t)]\mathbb{P}(\text{worker } w_{m_t} \text{ is available at } t) \\
&\overset{(5.30),(5.33)}{\geq} \sum_{t=l_0}^{T} \frac{1}{T} OPT \frac{1}{e} \\
&\to \frac{1}{e} OPT,
\end{aligned}$$

as $T \to +\infty$, which completes the proof. ■

In RSSAP, the reward for assigning a job to a worker is given by the product of the worker's success rate and the job value. However, the rolling window algorithm can be generalized to cases where rewards for assigning a job J to a work w_m is given by any function $r(J, w_m) = f(J, w_m)$, and Theorem 30 still holds.

5.3.3 Memoryless-length job sequences

This section considers memoryless-length job sequences: job lengths follow a geometric distribution with parameter q, independent of their values. Once a job is assigned to a worker, the worker becomes available at the beginning of the next time slot with probability q. Therefore, the expected length of a memoryless-length job is $\lceil 1/q \rceil$ (the ceiling is due to the discretized time axis). As in Section 5.3.2, we assume $m_0 = \lceil 1/q \rceil$ (if $m_0 \geq \lceil 1/q \rceil$, we use workers with the $\lceil 1/q \rceil$ largest success rates; otherwise, $m_0 < \lceil 1/q \rceil$, add $\lceil 1/q \rceil - m_0$ virtual workers with success rate zero).

We use the rolling window algorithm for memoryless-length jobs as follows: (1) job assignments start from time $\lceil 1/q \rceil$; (2) the length of each rolling window is $\lceil 1/q \rceil$, and hence the rolling window for job J_t arriving at time t is $W_t = [t - \lceil 1/q \rceil + 1, t]$, $t \geq \lceil 1/q \rceil$. Therefore, when a job J_t arrives at time $t \geq \lceil 1/q \rceil$, the rolling window algorithm computes the optimal weighted matching between jobs arriving in the rolling window $W_t = [t - \lceil 1/q \rceil + 1, t]$

and all workers $\{w_1, w_2, \ldots, w_{m_0}\}$. If the optimal weighted matching for W_t assigns job J_t to worker w_{m_t}, and worker w_{m_t} is available at time t, then the rolling window algorithm assigns job J_t to worker w_{m_t}; otherwise, discard job J_t. The length of each rolling window is the expected length of memoryless-length jobs, since each worker can complete at most one job in expectation within a rolling window, and hence the optimal matching (in expectation) between jobs arriving in this rolling window and workers is the same as the optimal schedule (in expectation) for these jobs on workers.

Theorem 31. *The rolling window algorithm is $e^2/(e-1)$-competitive for memoryless-length job sequences with a random order of arrivals.*

Proof: The proof follows an argument similar to that of Theorem 30. We consider the expected reward for an assignment given by the optimal weighted matching at each time $t \geq \lceil 1/q \rceil$ and the probability of such an assignment being feasible.

Let OPT denote the optimal offline reward for scheduling T jobs. (5.30) still holds for the expected reward for an assignment at time $t \geq \lceil 1/q \rceil$. Suppose that the optimal weighted matching between jobs arriving during rolling window $W_t = [t - \lceil 1/q \rceil + 1, t]$ and all workers assigns job J_t to worker w_{m_t}. We compute the probability of worker w_{m_t} being available at time t.

Similar to (5.31), the probability that worker w_{m_t} is assigned at some time $t' \in [t - \lceil 1/q \rceil + 1, t-1]$ has an upper bound

$$\begin{aligned}
&\mathbb{P}(\text{worker } w_{m_t} \text{ is assigned at time } t') \\
=&\mathbb{P}(\text{worker } w_{m_t} \text{ is matched to job } J_{t'})\mathbb{P}(\text{worker } w_{m_t} \text{ is available at } t') \\
\leq&\mathbb{P}(\text{worker } w_{m_t} \text{ is matched to job } J_{t'}) = \frac{1}{\lceil 1/q \rceil}. \qquad (5.34)
\end{aligned}$$

Since the job length follows a geometric distribution with parameter q, if worker w_{m_t} is not assigned in $[t - \lceil 1/q \rceil + 1, t-1]$ but assigned before rolling window W_t, there is still a probability that worker w_{m_t} has not completed the assigned job. Since the rolling window has length $\lceil 1/q \rceil$, then define

$$\begin{aligned}
P_1 \triangleq&\ \mathbb{P}\left(\text{worker } w_{m_t} \text{ is available before } t | \text{worker } w_{m_t} \text{ is assigned before } W_t\right) \\
\geq&\ 1 - (1-q)^{\lceil 1/q \rceil} \geq 1 - (1-q)^{1/q}. \qquad (5.35)
\end{aligned}$$

Let $Event_{W_t}$ denote the event that worker w_{m_t} is assigned before rolling

window W_t. Then

$$\begin{aligned}
&\mathbb{P}(\text{worker } w_{m_t} \text{is available at } t) \\
\geq &\mathbb{P}(\text{worker } w_{m_t} \text{is not assigned in } [t-\lceil 1/q \rceil + 1, t-1]) \\
&\times \Big(\mathbb{P}(\text{not } Event_{W_t}) \\
&\quad + \mathbb{P}(\text{worker } w_{m_t} \text{ is available before } t | Event_{W_t}) \mathbb{P}(Event_{W_t}) \Big) \quad (5.36) \\
\geq &\mathbb{P}(\text{worker } w_{m_t} \text{is not assigned in } [t-\lceil 1/q \rceil + 1, t-1]) P_1 \\
\geq &\left(\Pi_{t'=t-\lceil 1/q \rceil + 1}^{t-1} \mathbb{P}(\text{worker } w_{m_t} \text{is not assigned at } t') \right) P_1 \quad (5.37) \\
= &\left(\Pi_{t'=t-\lceil 1/q \rceil + 1}^{t-1} \left(1 - \mathbb{P}(\text{worker } w_{m_t} \text{is assigned at } t')\right) \right) P_1 \\
\geq &\left(\Pi_{t'=t-\lceil 1/q \rceil + 1}^{t-1} \left(1 - \frac{1}{\lceil 1/q \rceil} \right) \right) \left(1 - (1-q)^{\lceil 1/q \rceil}\right) \quad (5.38) \\
\geq &\frac{1}{e}(1 - \frac{1}{e}), \quad (5.39)
\end{aligned}$$

where: (a) inequality (5.36) follows from the law of total probability; (b) inequality (5.37) follows since the respective events that worker w_{m_t} is assigned at $t' \in [t - \lceil 1/q \rceil + 1, t-1]$ are independent; (c) inequality (5.38) follows from (5.34) and (5.35); (d) inequality (5.39) follows since $(1-q)^{\lceil 1/q \rceil} \leq (1-q)^{1/q} \leq 1/e$ for $0 < q \leq 1$.

Let $R_{RW}^{m_0}$ denote the reward for the rolling window algorithm for memoryless-length job sequences with a random order of arrivals. Combining (5.30) and (5.39) leads to

$$\begin{aligned}
\mathbb{E}[R_{RW}^{m_0}] &= \sum_{t=1}^{T} \mathbb{E}[\text{reward in time slot } t] \mathbb{P}(\text{worker } w_{m_t} \text{is available at } t) \\
&\geq \sum_{t=\lceil 1/q \rceil}^{T} \mathbb{E}[\text{reward in time slot } t] \mathbb{P}(\text{worker } w_{m_t} \text{is available at } t) \\
&\overset{(5.30),(5.39)}{\geq} \sum_{t=\lceil 1/q \rceil}^{T} \frac{1}{T} OPT \frac{1}{e}(1 - \frac{1}{e}) \\
&\to \frac{e-1}{e^2} OPT,
\end{aligned}$$

as $T \to +\infty$, which completes the proof. ■

Note that Theorem 31 holds for all memoryless-length jobs with $0 < q \leq 1$. For a specific value of q, a stronger competitive ratio $e/(1-(1-q)^{\lceil 1/q \rceil})$

can be computed from (5.38). When $q = 0$, the scheduling problem for memoryless-length jobs reduces to the weighted bipartite online matching problem. Note that $\mathbb{P}(Event_{W_t}) \to 0$ as $\lceil 1/q \rceil \to +\infty$, and the competitive ratio computed from (5.36) to (5.39) becomes e, which is consistent with the e-competitiveness given by [53].

5.3.4 General geometric and random order of arrivals

This section considers jobs with arrivals that are both geometric with $0 < p \leq 1$ and randomly ordered. Recall that for geometric arrivals, jobs arrive at the beginning of each time slot with probability $0 < p \leq 1$. The total number of jobs, T, is fixed and given. Sections 5.3.2 and 5.3.3 discuss the special case of geometric arrivals with $p = 1$ for equal-length and memoryless-length jobs, respectively. We extend those results to the more general geometric arrivals.

If there is no job arriving at the beginning of a time slot, we say a *virtual job* with value zero arrives. However, simply treating zero-value virtual jobs the same as real jobs will result in nonuniform order of job arrivals, illustrated by the following example.

Example 2. *Consider a sequence of equal-length jobs with $l_0 = 6$ and $p = 1/2$ for geometric arrivals. Consider the arrival order of real jobs and virtual jobs in a time interval $W = [1, 6]$. Then*

$$\mathbb{P}(2 \text{ real jobs arrive during } W) = \binom{6}{2} p^2 (1-p)^4 = \frac{6!}{2!4!} p^2 (1-p)^4.$$

Therefore, the probability for each arrival order of the two real jobs and four virtual jobs in this time interval W is

$$\begin{aligned} \mathbb{P}(\text{each arrival order for 2 real jobs in } W) &= \frac{\mathbb{P}(2 \text{ real jobs arrive during } W)}{\text{number of permutations}} \\ &= \frac{\frac{6!}{2!4!} p^2 (1-p)^4}{6!} = \frac{1}{4! 2^7}. \quad (5.40) \end{aligned}$$

Similarly,

$$\mathbb{P}(1 \text{ real job arrives during } W) = \binom{6}{1} p (1-p)^5 = \frac{6!}{5!} p (1-p)^5.$$

Therefore, the probability for each arrival order of the one real job and five

virtual jobs in this time interval W is

$$\mathbb{P}(\textit{each arrival order for 1 real job in } W) = \frac{\mathbb{P}(1 \textit{ real job arrives during } W)}{\textit{number of permutations}} = \frac{\frac{6!}{5!}p(1-p)^5}{6!} = \frac{1}{5!2^6}. \quad (5.41)$$

The probability of an arrival order with two real jobs and four virtual jobs arriving during W, (5.40), is not equal to the probability of an arrival order with one real job and five virtual jobs arriving during W, (5.41). ■

Consider a time interval W of length l. Then the number of arrivals of real jobs in W follows a binomial distribution with parameter l and p, and hence the expected number of arrivals of real jobs in the interval is $\mathbb{E}[N_W] = lp$. When the number of real job arrivals during a time interval is given, these real jobs can be modeled as uniformly and randomly selected from T real jobs.

We use the rolling window algorithm for equal-length jobs with general geometric and a random order of arrivals, where the size of rolling windows is l_0. As in Section 5.3.2, we further assume the number of workers is $m_0 = l_0$. Theorem 32 provides the competitive ratio of the rolling window algorithm for equal-length job sequences.

Theorem 32. *The rolling window algorithm is e^p-competitive for equal-length jobs with geometric and a random order of arrivals, where p is the parameter for geometric arrivals.*

Proof: The proof has the same structure as the proof of Theorem 30. We prove the e^p-competitiveness by considering the expected reward for an assignment given by the optimal weighted matching at time $t \geq l_0$ and the probability of such an assignment being feasible, conditioning on the event that a real job arrives at time t. Note that for geometric arrivals, a real job arrives at each time slot with probability p, and a virtual job with value zero arrives with probability $1 - p$.

Let OPT denote the optimal offline reward for scheduling T real jobs. Suppose that there is a real job J_t arriving at time t. Let $W_t = [t - l_0 + 1, t]$ denote the rolling window for assigning job J_t using the rolling window algorithm. Let N_W denote the number of real jobs arriving during $\hat{W}_t =$

$[t - l_0 + 1, t - 1]$. Then N_W is a binomial random variable with parameter $(l_0 - 1)$ and p, and $\mathbb{P}(N_W = x) = \binom{l_0-1}{x} p^x (1-p)^{l_0-x-1}$, $x = 0, 1, \ldots, l_0 - 1$.

Let $Event_{x,t}$ denote the event that a real job arrives at time t and x real jobs arrive during the rolling window W_t, $x = 1, 2, \ldots, l_0$. Then conditioning on the event that there are x real jobs arriving during W_t (including job J_t arriving at time t), these x jobs can be modeled as uniformly and randomly selected from T jobs due to the random order of arrivals, $x = 1, 2, \ldots, l_0$. Therefore, the conditional expected reward for scheduling jobs arriving during W_t given $Event_{x,t}$ has a lower bound

$$\mathbb{E}[R(W_t)|Event_{x,t}] \geq \frac{x}{T} OPT. \tag{5.42}$$

Moreover, the conditional expected reward for the assignment at time t is $\mathbb{E}[R(W_t)|Event_{x,t}]/x$, since the arrival order of these x jobs is a random permutation among themselves. Let $R(t)$ denote the reward for the assignment at time t. Then

$$\begin{aligned}
&\mathbb{E}[R(t)|\text{a job arrives at time } t] \\
=&\frac{\sum_{x=1}^{l_0} \mathbb{E}[R(t)|Event_{x,t}]\mathbb{P}(Event_{x,t})}{\mathbb{P}(\text{a job arrives at time } t)} \\
=&\frac{\sum_{x=1}^{l_0} \mathbb{E}[R(W_t)|Event_{x,t}] \frac{1}{x} \binom{l_0-1}{x-1} p^x (1-p)^{(l_0-x)}}{p} \\
\geq&\frac{OPT}{T}.
\end{aligned} \tag{5.43}$$

Consider the conditional probability of the assignment at time t being feasible, given that a real job J_t arrives at time t. Let w_{m_t} denote the worker assigned to job J_t under the optimal offline matching for rolling window W_t. Then the conditional probability of this assignment being feasible is the same as the conditional probability that worker w_{m_t} is available at time t. Since all jobs have the same length l_0, worker w_{m_t} is available at time t if the worker is not assigned to any other job during time interval $[t - l_0 + 1, t - 1]$. Note that given the total number of real job arrivals in a rolling window, the arrival order of these real jobs is a random permutation among themselves. Then, following a similar argument as that for (5.31), an upper bound for the conditional probability that worker w_{m_t} is assigned to a real job at time

$t' \in [t - l_0 + 1, t - 1]$ is

$$\begin{aligned}
&\mathbb{P}(\text{worker } w_{m_t} \text{ is assigned to a real job at } t' | \text{a job arrives at } t) \\
=&\mathbb{P}(\text{worker } w_{m_t} \text{ is assigned to a real job at } t') \\
\leq&\mathbb{P}(\text{worker } w_{m_t} \text{is assigned to a real or virtual job at } t') \\
=&\mathbb{P}(\text{worker } w_{m_t} \text{ is matched to a real or virtual job } J_{t'}) \\
&\times \mathbb{P}(\text{worker } w_{m_t} \text{is available at } t') \\
\leq&\mathbb{P}(\text{worker } w_{m_t} \text{ is matched to real or virtual job } J_{t'}) \\
=&\frac{1}{l_0}, \qquad (5.44)
\end{aligned}$$

where the first equality follows since the optimal weighted matching for rolling window $W_{t'} = [t' - l_0 + 1, t']$ is independent of the job arrival at

time $t > t'$. Then

$$
\begin{aligned}
&\mathbb{P}(\text{worker } w_{m_t} \text{ is available at } t|\text{a job arrives at } t) \\
=&\mathbb{P}(\text{worker } w_{m_t} \text{ is not assigned in } [t-l_0+1, t-1]|\text{a job arrives at } t) \\
=&\sum_{x=1}^{l_0} \mathbb{P}(\text{worker } w_{m_t} \text{ is not assigned in } [t-l_0+1, t-1]|Event_{x,t}) \\
&\times \mathbb{P}((x-1) \text{ jobs arrive during } [t-l_0+1, t-1]|\text{a job arrives at } t) \qquad (5.45) \\
=&\sum_{x=1}^{l_0} \mathbb{P}(\text{worker } w_{m_t} \text{ is not assigned in } [t-l_0+1, t-1]|Event_{x,t}) \\
&\times \mathbb{P}((x-1) \text{ jobs arrive during } [t-l_0+1, t-1]) \qquad (5.46) \\
=&\sum_{x=1}^{l_0} \mathbb{P}(\text{worker } w_{m_t} \text{ is not assigned in } [t-l_0+1, t-1]|Event_{x,t}) \\
&\times \binom{l_0-1}{x-1} p^{x-1}(1-p)^{l_0-x} \qquad (5.47) \\
\geq&\sum_{x=1}^{l_0} \left(\Pi_{t'=t-l_0+1}^{t-1} \mathbb{P}(\text{worker } w_{m_t} \text{ is not assigned at } t'|Event_{x,t})\right) \\
&\times \binom{l_0-1}{x-1} p^{x-1}(1-p)^{l_0-x} \\
=&\sum_{x=1}^{l_0} \left(\Pi_{t'=t-l_0+1}^{t-1} \left(1-\mathbb{P}(\text{worker } w_{m_t} \text{ is assigned at } t'|Event_{x,t})\right)\right) \\
&\times \binom{l_0-1}{x-1} p^{x-1}(1-p)^{l_0-x} \\
\geq&\sum_{x=1}^{l_0} (1-\frac{1}{l_0})^{x-1} \binom{l_0-1}{x-1} p^{x-1}(1-p)^{l_0-x} \qquad (5.48) \\
=&(1-p+p(1-\frac{1}{l_0}))^{l_0-1} \\
=&(1-\frac{1}{l_0/p})^{(l_0/p-1)(p\frac{l_0-1}{l_0-p})} \\
\geq&e^{-p}, \qquad (5.49)
\end{aligned}
$$

where: (a) equality (5.45) follows from the law of total probability; (b) equality (5.46) follows since the events of $(x-1)$ real jobs arriving during $[t-l_0+1, t-1]$ and a job arriving at time t are independent for geometric arrivals (given that there are more jobs left to arrive); (c) equality (5.47) fol-

lows from the fact that the number of real job arrivals during $[t-l_0+1, t-1]$ is a binomial random variable with parameters (l_0-1) and p; (d) inequality (5.48) follows from (5.44) and worker w_{m_t} has $(x-1)$ opportunities to be assigned to one of the $(x-1)$ jobs arriving during $[t-l_0+1, t-1]$; (e) inequality (5.49) follows since $f(y) = (1-1/y)^{y-1} \geq e^{-1}$ for $y \geq 2$ and $p\frac{l_0-1}{l_0-p} \leq p$ for $0 < p \leq 1$.

Let R_{RW}^{Ep} denote the reward for the rolling window algorithm for equal-length jobs following geometric with parameter p and a random order of arrivals. Let T_p denote the time slot when the last real job arrives. Since the inter-arrival time between subsequent real jobs is a geometric random variable with parameter p, T_p is a random variable with $\mathbb{E}[T_p] = T\lceil 1/p \rceil$. Combining (5.43) and (5.49) leads to

$$
\begin{aligned}
&\mathbb{E}[R_{RW}^{Ep}] \\
=&\mathbb{E}\Bigg[\sum_{t=1}^{T_p} \mathbb{E}[R(t)|\text{a job arrives at } t] \\
&\times \mathbb{P}(\text{worker } w_{m_t} \text{is available at } t|\text{a job arrives at } t) \\
&\times \mathbb{P}(\text{a job arrives at } t)\Bigg] \\
\geq&\mathbb{E}\Bigg[\sum_{t=l_0}^{T_p} \mathbb{E}[R(t)|\text{a job arrives at } t] \\
&\times \mathbb{P}(\text{worker } w_{m_t} \text{is available at } t|\text{a job arrives at time } t) \\
&\times \mathbb{P}(\text{a job arrives at } t)\Bigg] \\
\overset{(5.43),(5.49)}{\geq}& \mathbb{E}\left[\sum_{t=l_0}^{T_p} \frac{OPT}{T} e^{-p} p\right] \\
=&\frac{OPT}{T} e^{-p} p(\mathbb{E}[T_p] - l_0 + 1) \\
\to&\frac{OPT}{e^p},
\end{aligned}
$$

as $T \to +\infty$, which completes the proof. ■

For memoryless-length jobs, we use the rolling window algorithm with rolling windows of size $\lceil 1/q \rceil$. As in Section 5.3.3, we further assume the number of workers is $m_0 = \lceil 1/q \rceil$. Theorem 33 provides the competitive ratio of the rolling window algorithm for memoryless-length jobs following

geometric and a random order of arrivals.

Theorem 33. *The rolling window algorithm is $e^{p+1}/(e-1)$-competitive for memoryless-length jobs following geometric and a random order of arrivals, where p is the parameter for geometric arrivals.*

Proof: See Appendix B. ■

CHAPTER 6

PRIMAL-DUAL APPROACH TO ONLINE INTERVAL SCHEDULING

This chapter provides a primal-dual approach to analyze algorithms for online interval scheduling problems. This primal-dual technique can be used for both stochastic and adversarial job sequences, and hence is universally and generally applicable. We use strong duality and complementary slackness conditions to derive optimal algorithms for scheduling stochastic job sequences on a single machine. We use weak duality to obtain upper bounds for the optimal reward for scheduling stochastic job sequences on multiple machines. We use the primal-dual technique to compute competitive ratios of an approximation algorithm for adversarial job sequences.

6.1 Formulation and primal-dual techniques

This section formulates online interval scheduling problems as a linear program and provides the primal-dual analysis framework.

Consider a sequence of N jobs $\mathbf{I} = \{J_1, J_2, \ldots, J_N\}$ to be scheduled on m machines, denoted by $M_1, M_2, \ldots, M_m$. A sequence of jobs is referred to as a *job instance*, denoted by $\mathbf{I}$. The scheduling of a job on a machine is referred to as an *assignment*. Each machine can process at most one job at a time and each job can be scheduled on at most one machine. Each machine has a given weight, denoted by $\{w_1, w_2, \ldots, w_m\}$. Without loss of generality, we assume $w_1 \leq w_2 \leq \ldots \leq w_m$. The complete job instance is not known beforehand (the total number of jobs N is not known as well), and jobs are revealed only at each arrival. Each assignment is made *online* upon each arrival.

Each job is featured by a *job vector*, revealed upon arrival. Let $J_j = (a_j, l_j, v_j)$ denote the job vector of the j^{th} job arrival, for $j = 1, 2, \ldots, N$, where a_j denotes the job *arrival time*, l_j denotes the job *length*, and v_j denotes

the job *value*. Furthermore, we assume that the time axis is discretized into *slots*, indexed by $t = 1, 2, \ldots, T$, such that job arrivals and completions only occur at the beginning and the end of time slots, respectively. Therefore, $a_j,\ l_j \in \mathbf{Z}^+$ for $j = 1, 2, \ldots, N$. Moreover, we assume that no jobs share the same arrival time. The *completion time* of job J_j is defined as $f_j \equiv a_j + l_j$. We refer to the half-open interval $[a_j, f_j)$ as the *job interval*. If two jobs J_{j_1} and J_{j_2} satisfy $[a_{j_1}, f_{j_1}) \bigcap [a_{j_2}, f_{j_2}) \neq \emptyset$, then J_{j_1} and J_{j_2} are said to *conflict* with each other.

We consider two cases of online interval scheduling problems:

SE: stochastic equal-length arbitrary-value jobs on multiple machines. There is one job arriving at the beginning of each time slot (since the case of no jobs arriving at a time slot can be seen as a job with value zero arriving at the slot). Jobs in an instance share the same length with values following a given distribution. Machines have different weights and the reward for completing a job J_j on machine M_i is given by $v_j \times w_i$. Job assignments are assumed to be *non-preemptive*, which requires a job scheduled on a machine to be completed without interruption.

AC: adversarial C-benevolent jobs on a single machine. There is not necessarily one job arriving at the beginning of each time slot. Jobs in an instance have different lengths with values as a function of lengths $v = v(l)$ (with a slight abuse of notation, we use v to denote both the job value and the job value as a function of the job length), where $v : \mathbf{R}^+ \to \mathbf{R}^+$ is nondecreasing, positive, and convex. The weight for the single machine is set as $w_1 = 1$, and hence the reward for completing a job is the job value. Job assignments are assumed to be *preemptive*, and hence a job assigned to a machine can be terminated before completion in favor of a later arriving job. However, the terminated job cannot be reassigned and its value is lost. We refer to such assignments as *temporary* assignments and terminated jobs as *aborted* jobs.

Let the binary variable $X_{i,j}$ denote the assignment of job J_j on machine M_i, for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, N$: if job J_j is assigned to machine M_i, $X_{i,j} = 1$; otherwise, $X_{i,j} = 0$. The objective of online interval scheduling problems is to maximize the total reward for completed jobs, subject to the constraint determined by the number of available machines. We formulate

both SE and AC as linear programs. Note that the constraints for the linear program are dynamic, and hence we consider the constraint for each time slot $t = 1, 2, \ldots, T$, where T (random variable) denotes the completion time of the last job in a job instance.

$$\max \quad \mathbb{E}[\sum_{i=1}^{m}\sum_{j=1}^{N} v_j w_i X_{i,j}], \tag{6.1}$$

$$s.t. \sum_{j:t\in[a_j,f_j)} X_{i,j} \leq 1, \qquad \text{for } i = 1, 2 \ldots, m \text{ and for all } t = 1, 2, \ldots, T, \tag{6.2}$$

$$\sum_{i=1}^{m} X_{i,j} \leq 1, \qquad \text{for } j = 1, 2, \ldots, N, \tag{6.3}$$

$$X_{i,j} \in \{0, 1\}, \qquad \text{for } i = 1, 2, \ldots, m \text{ and } j = 1, 2, \ldots, N. \tag{6.4}$$

Constraint (6.2) means that each machine can be assigned at most one job at any time slot t. Constraint (6.3) means that each job can be assigned to at most one machine. Constraint (6.4) means that each assignment is binary valued. The expectation is taken with respect to the distribution of job values (for stochastic online interval scheduling problems only) and the random assignments of the algorithm (for randomized algorithms only). The corresponding dual programs are given in Sections 6.2 and 6.3.

6.1.1 Primal-dual techniques

The primal-dual analysis framework for both SE and AC uses strong and weak duality. However, there are differences between the analysis approaches for these two cases since SE and AC consider the expected and the *worst-case* performance of an algorithm, respectively.

For SE, we construct a feasible dual solution and give the corresponding primal solution (if obtainable). The objective value of the dual program (referred to as *the value of the dual program*) based on the feasible dual solution provides an upper bound for the objective value of the primal program (referred to as *the value of the primal program*), which is the optimal reward for the scheduling problem. When the feasible solution to the dual program and the corresponding solution to the primal program satisfy the complementary

slackness conditions, we can prove that the corresponding solution to the primal program is an optimal algorithm to the scheduling problem.

For AC, since the adversarial job sequence is considered, we use *competitive ratios* to evaluate the worst-case performance of algorithms, as defined in Definition 5. Let $R_{\mathcal{A}}(\mathbf{I})$ and $OPT(\mathbf{I})$ denote the reward for algorithm $\mathcal{A}$ and the optimal reward for a job instance $\mathbf{I}$, respectively.

Definition 5. *A deterministic (randomized) algorithm $\mathcal{A}$ is said to have a competitive ratio γ if $R_{\mathcal{A}}(\mathbf{I}) \geq OPT(\mathbf{I})/\gamma$ ($\mathbb{E}[R_{\mathcal{A}}(\mathbf{I})] \geq OPT(\mathbf{I})/\gamma$, where the expectation is taken with respect to the random assignments made by $\mathcal{A}$) for any job instance $\mathbf{I}$.*

Job assignments are preemptive, and hence a previously assigned job may be terminated before completion in favor of a later arriving job. To prove an algorithm to be γ-competitive, we show the reward of the algorithm is at least $1/\gamma$ of the optimal reward for all possible job instances. A feasible scheduling algorithm always results in a feasible solution to the primal program, and the corresponding value of the primal program is the same as the reward for the algorithm. We construct a feasible solution to the dual program, with the value of the dual program no greater than γ times the value of the primal program. From weak duality, the optimal reward has an upper bound given by the value of the dual program. Therefore, the algorithm has a competitive ratio of γ.

The construction of a feasible solution to the dual program for AC depends on the specific job instance, and hence there is no general dual solution that is feasible for all possible job instances. We turn to constructing a dual solution that is feasible for the worst-case job instance, which is sufficient for computing the competitive ratio. For randomized algorithms, we follow the same framework described above for deterministic algorithms. The only differences are that: (1) we compare the expected reward of a randomized algorithm to the optimal reward; (2) we construct a solution to the dual program, which is feasible in expectation. Our techniques for randomized algorithms are motivated by [60].

6.2 Stochastic Online Interval Scheduling Problems

This section uses primal-dual techniques to solve stochastic online interval scheduling problems, where jobs have equal lengths with values following a given distribution (referred to as equal-length jobs). We provide a randomized optimal algorithm on a single machine using strong duality and complementary slackness conditions. We also provide an upper bound for the optimal expected reward on multiple machines using weak duality. For stochastic C-benevolent jobs, we provide an upper bound for the optimal expected reward on a single machine, which is tight when values of C-benevolent jobs are linearly proportional to their lengths.

Note that the value of (6.1) needs to be finite for strong duality to hold. Therefore, we consider the *stationary expected reward* for an algorithm, which is the expected reward per job as the total number of jobs N goes to infinity, computed by

$$\frac{1}{N}\mathbb{E}[\sum_{i=1}^{m}\sum_{j=1}^{N} v_j w_i X_{i,j}], \tag{6.5}$$

with the job value distribution satisfying $\mathbb{E}[v] < +\infty$. We assume that the job values are discrete with support $0 \leq V_1 < V_2 < \ldots < V_L < +\infty$. Let h_v denote the probability mass at job value v, for $v = V_1, V_2, \ldots, V_L$. We further assume the total number of jobs is unknown but fixed and sufficiently large.

6.2.1 SE on a single machine

This section considers SE on a single machine, i.e., $m = 1$. To simplify notations, we eliminate the subscript of i in all our variables and assume

$w_1 = 1$. Therefore, the objective function (6.5) reduces to

$$\begin{aligned}
&\frac{1}{N}\mathbb{E}[\sum_{j=1}^{N} v_j X_j] \\
=&\frac{1}{N}\sum_{j=1}^{N}\mathbb{E}[v_j X_j] \\
=&\frac{1}{N}\sum_{j=1}^{N}\sum_{v=V_1}^{V_L} v\mathbb{P}(v_j = v)\mathbb{P}(X_j = 1|v_j = v) \qquad (6.6)\\
\equiv&\frac{1}{N}\sum_{j=1}^{N}\sum_{v=V_1}^{V_L} v h_v p_{j|v},
\end{aligned}$$

where (6.6) follows from the law of total probability. The probability $p_{j|v} \equiv \mathbb{P}(X_j = 1|v_j = v)$ is the assignment variable for a *randomized* algorithm: given $v_j = v$, the algorithm schedules job J_j with probability $p_{j|v}$.

We simplify constraints (6.2) to (6.4) for scheduling stochastic equal-length jobs on a single machine. For SE, we assume that there is a job arriving at each time slot t. Therefore, the arrival time of the j^{th} job is in time slot j with $a_j = j$, and hence, t and j are equivalent notations. We can then simplify the notations by using j for both the arrival order and the arrival time of the j^{th} job. Let $q_{j,v} \equiv h_v p_{j|v}$ and hence $q_{j,v} \leq h_v$. Let $l_0 > 1$ denote the length of all equal-length jobs. We generalize constraint (6.2) of the primal program for randomized algorithms by taking expectations on both sides,

$$\mathbb{E}[\sum_{j:t\in[a_j,f_j)} X_j] = \mathbb{E}[\sum_{j:j'\in[a_j,f_j)} X_j] = \mathbb{E}[\sum_{j:j'\in[j,j+l_0)} X_j] \qquad (6.7)$$

$$\begin{aligned}
&= \mathbb{E}[\sum_{j=j'-l_0+1}^{j'} X_j] \\
&= \sum_{j=j'-l_0+1}^{j'} \mathbb{E}[X_j] \qquad (6.8)\\
&= \sum_{j=j'-l_0+1}^{j'}\sum_{v=V_1}^{V_L} h_v\mathbb{P}(X_j = 1|v_j = v) \\
&= \sum_{j=j'-l_0+1}^{j'}\sum_{v=V_1}^{V_L} q_{j,v} \leq 1,
\end{aligned}$$

for $j' = 1, 2, \ldots, N$, where (6.7) follows from $a_j = j$ and $f_j = j + l_0$ for all j, and t and j are equivalent notations; (6.8) follows from the linearity of expectations and no randomness in the total number of terms contained in the summation of $\sum_{j=j'-l_0+1}^{j'} X_j$. Constraint (6.3) is not necessary for a single machine since $m = 1$. For randomized algorithms, constraint (6.4) is given in the form of $p_{j|v} \leq 1$, and hence $q_{j,v} \leq h_v$. Therefore, the primal linear program for SE on a single machine is

$$\max \quad \frac{1}{N} \sum_{j=1}^{N} \sum_{v=V_1}^{V_L} v q_{j,v}, \tag{P1}$$

$$s.t. \sum_{j=j'-l_0+1}^{j'} \sum_{v=V_1}^{V_L} q_{j,v} \leq 1, \quad \text{for } j' = 1, 2, \ldots, N, \tag{6.9}$$

$$q_{j,v} \leq h_v, \qquad \text{for } j = 1, 2, \ldots, N \text{ and } v = V_1, V_2, \ldots, V_L. \tag{6.10}$$

The corresponding dual linear program is

$$\min \quad \sum_{j=1}^{N} \alpha_j + \sum_{j=1}^{N} \sum_{v=V_1}^{V_L} \beta_{j,v} h_v, \tag{D1}$$

$$s.t. \sum_{j'=j}^{j+l_0-1} \alpha_{j'} + \beta_{j,v} \geq \frac{v}{N}, \text{ for } j = 1, 2, \ldots, N, \text{ and } v = V_1, V_2, \ldots, V_L. \tag{6.11}$$

The dual variables $\{\alpha_j\}$ and $\{\beta_{j,v}\}$ correspond to constraints (6.9) and (6.10), respectively. The variable α_j denotes the *basic cost* for time slot j. The variable $\beta_{j,v}$ denotes the *additional cost* for a job arriving in time slot j with value v. The complementary slackness conditions for the optimal solutions

to the primal and dual programs are given by

$$\alpha_{j'}(\sum_{j=j'-l_0+1}^{j'}\sum_{v=V_1}^{V_L} q_{j,v}-1)=0, \quad \text{for } j'=1,2,\ldots,N, \tag{6.12}$$

$$\beta_{j,v}(q_{j,v}-h_v)=0, \quad \text{for } j=1,2,\ldots,N \text{ and } v=V_1,V_2,\ldots,V_L, \tag{6.13}$$

$$q_{j,v}(\sum_{j'=j}^{j+l_0-1}\alpha_{j'}+\beta_{j,v}-\frac{v}{N})=0, \quad \text{for } j=1,2,\ldots,N \text{ and } v=V_1,V_2,\ldots,V_L. \tag{6.14}$$

Theorem 34 provides an upper bound for the optimal expected reward for scheduling stochastic equal-length jobs on a single machine.

Theorem 34. *An upper bound for the optimal expected reward for scheduling IID equal-length jobs on a single machine is*

$$\left(\frac{1}{l_0}-\mathbb{P}(v\geq V_{s-1})\right)V_{s-1}+\mathbb{E}[v|v\geq V_{s-1}]\mathbb{P}(V\geq V_{s-1}).$$

This upper bound is achieved under an optimal randomized policy, which is feasible in expectation, given by

$$q_{j,v}=\begin{cases} h_v, \text{ for } v\geq V_s, \\ \frac{1}{l_0}-\sum_{v'=V_s}^{V_L}h_{v'}, \text{ for } v=V_{s-1}, \\ 0, \text{ for } v<V_{s-1}, \end{cases} \tag{6.15}$$

for all $j=1,2,\ldots,N$, where $V_s\in\{V_2,V_3,\ldots,V_L\}$ satisfies $\mathbb{P}(v>V_s)\leq 1/l_0<\mathbb{P}(v>V_{s-1})$.

Proof: We first construct feasible solutions $\{q^*_{j,v}\}$ and $\{\alpha^*_j\}$, $\{\beta^*_{j,v}\}$ for (P1) and (D1), respectively. Then we show that they satisfy the complementary slackness conditions (6.12) to (6.14), which proves that they are optimal solutions to the corresponding linear optimization problems (P1) and (D1), respectively.

There exists $V_s\in\{V_2,V_3,\ldots,V_L\}$ such that $\mathbb{P}(v>V_s)\leq 1/l_0<\mathbb{P}(v>V_{s-1})$. We construct a dual solution as follows:

$$\alpha_j=\alpha=\frac{V_{s-1}}{Nl_0}, \text{ for all } j=1,2,\ldots,N, \tag{6.16}$$

and

$$\beta_{j,v} = \beta_v = \begin{cases} (v - V_{s-1})/N, \text{ for } V \geq V_{s-1}, \\ 0, \text{ for } V < V_{s-1}, \end{cases} \tag{6.17}$$

for all $j = 1, 2, \ldots, N$. To see that the solution given by (6.16) and (6.17) is feasible,

$$\sum_{j'=j}^{j+l_0-1} \alpha_{j'} + \beta_{j,v} = l_0\alpha + \beta_v = \begin{cases} \frac{V_{s-1}}{N} + (v - V_{s-1})/N = \frac{v}{N}, & \text{for } V \geq V_{s-1}, \\ \frac{V_{s-1}}{N}, & \text{for } V < V_{s-1}, \end{cases}$$

and hence constraint (6.11) is satisfied. Therefore, the dual solution given by (6.16) and (6.17) is a feasible dual solution.

The value of the dual program (D1) becomes

$$\begin{aligned} & N\alpha + N\sum_{v=V_1}^{V_L} \beta_v h_v \\ \overset{(6.16),(6.17)}{=} & N\frac{V_{s-1}}{Nl_0} + N\sum_{v=V_{s-1}}^{V_L} \frac{v - V_{s-1}}{N} h_v \\ = & \left(\frac{1}{l_0} - \mathbb{P}(v \geq V_{s-1})\right) V_{s-1} + \mathbb{E}[v | v \geq V_{s-1}]\mathbb{P}(V \geq V_{s-1}). \end{aligned} \tag{6.18}$$

We are left to show that the dual solution (6.16) and (6.17) and the primal solution (6.15) satisfy the complementary slackness conditions (6.12) to (6.14). For (6.12), since $\alpha_j = \alpha > 0$ for all j, then

$$\alpha_{j'}\Big(\sum_{j=j'-l_0+1}^{j'} \sum_{v=V_1}^{V_L} q_{j,v} - 1\Big) \overset{(6.15)}{=} \alpha_{j'}\Big(l_0\Big(\sum_{v=V_s}^{V_L} h_v + \frac{1}{l_0} - \sum_{v=V_s}^{V_L} h_v\Big)\Big) = 0,$$

for $j' = 1, 2, \ldots, N$. For (6.13), note that $\beta_{j,v} = 0$ for $v \leq V_{s-1}$ and $j = 1, 2, \ldots, N$; $q_{j,v} = h_v$ for $v \geq V_s$ and $j = 1, 2, \ldots, N$, and hence

$$\beta_{j,v}(q_{j,v} - h_v) = 0.$$

For (6.14), note that $q_{j,v} = 0$ for $v < V_{s-1}$ and $j = 1, 2, \ldots, N$; for $v \geq V_{s-1}$

and $j = 1, 2, \ldots, N$,

$$q_{j,v}(\sum_{j'=j}^{j+l_0-1} \alpha_{j'} + \beta_{j,v} - \frac{v}{N}) \overset{(6.16),(6.17)}{=} q_{j,v}(\frac{V_{s-1}}{N} + \frac{v - V_{s-1}}{N} - \frac{v}{N}) = 0.$$

■

6.2.2 C-benevolent jobs on a single machine

This section considers C-benevolent jobs, whose values are subject to a function of job lengths, $v = v(l)$. The function $v : \mathbf{R}^+ \to \mathbf{R}^+$ is non-decreasing, positive and convex. As in Section 6.2.1, we assume that the job values are discrete and let h_v denote the probability mass for job value $v = V_1, V_2, \ldots, V_L$. Let l_v and l^j denote the length of a job with value $v = V_1, V_2, \ldots, V_L$, and the length of the job arriving at time $j = 1, 2, \ldots, N$ (i.e., the j^{th} arrived job).

For C-benevolent jobs, the objective functions of the primal and the dual programs remain the same as (D1) and (P1). However, the constraint (6.2) for the primal program changes to

$$\mathbb{E}[\sum_{j:t\in[a_j,f_j)} X_j] = \mathbb{E}[\sum_{j:j'\in[a_j,f_j)} X_j] = \mathbb{E}[\sum_{j:j'\in[j,j+l^j)} X_j] \tag{6.19}$$

$$= \sum_{v=V_1}^{V_L} \sum_{j=j'-l_v+1}^{j'} h_v \mathbb{P}(X_j = 1 | v_j = v) \tag{6.20}$$

$$= \sum_{v=V_1}^{V_L} \sum_{j=j'-l_v+1}^{j'} q_{j,v} \leq 1, \tag{6.21}$$

for all $j' = 1, 2, \ldots, N$, where (6.19) follows from $a_j = j$ and $f_j = a_j + l^j$, and t and j are equivalent notations; (6.20) follows from the law of total probability and l^j is a random variable with probability mass h_v for job length l_v, $v = V_1, V_2, \ldots, V_L$. The constraint for the corresponding dual program is

$$\sum_{j'=j}^{j+l_v-1} \alpha_{j'} + \beta_{j,v} \geq \frac{v}{N}, \text{ for } j = 1, 2, \ldots, N, \text{ and } v = V_1, V_2, \ldots, V_L. \tag{6.22}$$

Then the complementary slackness conditions are

$$\alpha_{j'}(\sum_{v=V_1}^{V_L}\sum_{j=j'-l_v+1}^{j'} q_{j,v}-1)=0, \quad \text{for } j'=1,2,\ldots,N, \tag{6.23}$$

$$\beta_{j,v}(q_{j,v}-h_v)=0, \quad \text{for } j=1,2,\ldots,N \text{ and } v=V_1,V_2,\ldots,V_L, \tag{6.24}$$

$$q_{j,v}(\sum_{j'=j}^{j+l_v-1}\alpha_{j'}+\beta_{j,v}-\frac{v}{N})=0, \quad \text{for } j=1,2,\ldots,N \text{ and } v=V_1,V_2,\ldots,V_L. \tag{6.25}$$

We construct a feasible dual solution, which gives an upper bound for the optimal expected reward. When the C-benevolent function is linearly proportional (i.e., $v=\gamma l$, where γ is a positive constant), the dual solution has a corresponding primal solution, and the complementary slackness conditions are satisfied. Therefore, the primal solution is the optimal solution to the primal program.

Theorem 35 provides an upper bound for the optimal expected reward for scheduling stochastic C-benevolent jobs on a single machine.

Theorem 35. *An upper bound for the optimal expected reward for scheduling IID C-benevolent jobs on a single machine is*

$$\frac{V_{s-1}}{l_{V_{s-1}}}\left(1-\mathbb{E}[l|v\geq V_s]\mathbb{P}(v\geq V_s)\right)+\mathbb{E}[v|v\geq V_s]\mathbb{P}(v\geq V_s), \tag{6.26}$$

where $V_s\in\{V_2,V_3,\ldots,V_L\}$ *satisfies* $\mathbb{E}[l|v\geq V_s]\mathbb{P}(v\geq V_s)<1\leq\mathbb{E}[l|v\geq V_{s-1}]\mathbb{P}(v\geq V_{s-1})$.

Proof:

Note that $\mathbb{E}[l|v\geq V_{th}]\mathbb{P}(v\geq V_{th})=\sum_{v=V_{th}}^{V_L}l_vh_v$ is a decreasing function of V_{th}, and $\mathbb{E}[l|v\geq 0]\mathbb{P}(v\geq 0)=\mathbb{E}[l]>1$. Then there exists a $V_s\in\{V_2,V_3,\ldots,V_L\}$ such that $\mathbb{E}[l|v\geq V_s]\mathbb{P}(v\geq V_s)<1\leq\mathbb{E}[l|v\geq V_{s-1}]\mathbb{P}(v\geq V_{s-1})$. We construct a dual solution as

$$\alpha_j=\alpha=\frac{V_{s-1}}{Nl_{V_{s-1}}}, \tag{6.27}$$

and

$$\beta_{j,v} = \beta_v = \begin{cases} 0, & \text{for } V \le V_{s-1}, \\ \frac{v}{N} - l_v \frac{V_{s-1}}{N l_{V_{s-1}}}, & \text{for } V \ge V_s, \end{cases} \tag{6.28}$$

for all $j = 1, 2, \ldots, N$. To see the dual solution given by (6.27) and (6.28) is feasible solution, we show constraint (6.22) is satisfied,

$$\sum_{j'=j}^{j+l_v-1} \alpha_{j'} + \beta_{j,v} = \begin{cases} \frac{V_{s-1}}{N l_{V_{s-1}}} l_v \ge \frac{v}{N}, & \text{for } V \le V_{s-1}, \\ \frac{V_{s-1}}{N l_{V_{s-1}}} l_v + \frac{v}{N} - l_v \frac{V_{s-1}}{N l_{V_{s-1}}} \ge \frac{v}{N}, & \text{for } V \ge V_s, \end{cases}$$

which follows from the property of C-benevolent jobs: l_v is an increasing function of v and v/l_v is an increasing function of v.

The value of the dual program has an upper bound given by the feasible dual solution (6.27) and (6.28)

$$\begin{aligned} & N \frac{V_{s-1}}{N l_{V_{s-1}}} + N \sum_{v=V_s}^{V_L} (\frac{v}{N} - \frac{V_{s-1}}{N l_{V_{s-1}}} l_v) h_v \\ =& \frac{V_{s-1}}{l_{V_{s-1}}} + \sum_{v=V_s}^{V_L} v h_v - \frac{V_{s-1}}{l_{V_{s-1}}} \sum_{v=V_s}^{V_L} l_v h_v \\ =& \frac{V_{s-1}}{l_{V_{s-1}}} \left(1 - \mathbb{E}[l | v \ge V_s] \mathbb{P}(v \ge V_s)\right) + \mathbb{E}[v | v \ge V_s] \mathbb{P}(v \ge V_s), \end{aligned}$$

which gives an upper bound for the optimal expected reward for scheduling C-benevolent jobs, from weak duality. ■

For special cases of C-benevolent jobs, stronger results can be obtained. Corollary 6 provides the optimal solution for scheduling proportional-value C-benevolent jobs on a single machine, which is a randomized algorithm that is feasible in expectation.

Corollary 6. *When the C-benevolent jobs satisfy $v = v(l) = \gamma l$, where $\gamma > 0$ is a constant, the optimal solution (feasible in expectation) for scheduling IID C-benevolent jobs on a single machine is given by*

$$q_{j,v} = \frac{h_v}{l_v}, \text{ for all } j = 1, 2, \ldots N, \text{ and } v = V_1, V_2, \ldots, V_L. \tag{6.29}$$

Proof:

When the C-benevolent jobs satisfy $v = v(l) = \gamma l$, where $\gamma > 0$ is a

constant, a solution to the dual program is as follows:

$$\begin{aligned} \alpha_j &= \frac{\gamma}{N}, \text{ for all } j = 1, 2, \ldots, N, \\ \beta_v &= 0, \text{ for all } v = V_1, V_2, \ldots, V_L. \end{aligned} \tag{6.30}$$

To see the solution given by (6.30) is a feasible solution to the dual program, we show constraint (6.22) is satisfied

$$\sum_{j'=j}^{j+l_v-1} \alpha_{j'} + \beta_{j,v} = \frac{\gamma l_v}{N} = \frac{v}{N},$$

for $j = 1, 2, \ldots, N$ and $v = V_1, V_2, \ldots, V_L$.

We prove that the solutions given by (6.29) and (6.30) satisfy the complementary slackness conditions (6.23) to (6.25), and hence that they are the optimal solutions to the primal and dual programs. To see this,

$$\begin{aligned} \alpha_{j'} \left(\sum_{v=V_1}^{V_L} \sum_{j=j'-l_v+1}^{j'} q_{j,v} - 1 \right) &= \frac{\gamma}{N} \left(\sum_{v=V_1}^{V_L} \sum_{j=j'-l_v+1}^{j'} \frac{h_v}{l_v} - 1 \right) \\ &= \frac{\gamma}{N} \left(\sum_{v=V_1}^{V_L} h_v - 1 \right) = 0, \end{aligned}$$

for $j' = 1, 2, \ldots, N$.

$$\beta_{j,v}(q_{j,v} - h_v) = 0,$$

for $j = 1, 2, \ldots, N$ and $v = V_1, V_2, \ldots, V_L$.

$$q_{j,v} \left(\sum_{j'=j}^{j+l_v-1} \alpha_{j'} + \beta_{j,v} - \frac{v}{N} \right) = q_{j,v} \left(\frac{l\gamma}{N} - \frac{v}{N} \right) = 0,$$

for $j = 1, 2, \ldots, N$ and $v = V_1, V_2, \ldots, V_L$. ■

6.2.3 SE on multiple machines

This section extends the results in Section 6.2.1 to multiple machines with different weights. Suppose that the weights of m machines satisfy $w_1 \leq w_2 \leq \ldots \leq w_m$. The objective function of the primal program is to maximize the

expected reward per job:

$$\begin{aligned}
&\frac{1}{N}\mathbb{E}[\sum_{i=1}^{m}\sum_{j=1}^{N} X_{i,j}w_i v_j] \\
=&\frac{1}{N}\sum_{i=1}^{m}\sum_{j=1}^{N}\mathbb{E}[X_{i,j}w_i v_j] \\
=&\frac{1}{N}\sum_{i=1}^{m}\sum_{j=1}^{N}\sum_{v=V_1}^{V_L} w_i v h_v \mathbb{P}(X_{i,j}=1|v_j=v) \\
\equiv&\frac{1}{N}\sum_{i=1}^{m}\sum_{j=1}^{N}\sum_{v=V_1}^{V_L} w_i v h_v p_{i,j,v},
\end{aligned}$$

where h_v denotes the probability mass at job value $v = V_1, V_2, \ldots, V_L$, and $p_{i,j|v} \equiv \mathbb{P}(X_{i,j} = 1|v_j = v)$ is the assignment variable for a randomized algorithm: given $v_j = v$, the algorithm assigns job J_j to machine M_i with probability $p_{i,j|v}$. Similarly, we can write constraint (6.2) for randomized algorithms as

$$\begin{aligned}
\mathbb{E}[\sum_{j:t\in[a_j,f_j)} X_{i,j}] = \mathbb{E}[\sum_{j:j'\in[a_j,f_j)} X_{i,j}] &= \mathbb{E}[\sum_{j:j'\in[j,j+l_0)} X_{i,j}] \\
&= \sum_{j=j'-l_0+1}^{j'} \mathbb{E}[X_{i,j}] \\
&= \sum_{j=j'-l_0+1}^{j'}\sum_{v=V_1}^{V_L} h_v \mathbb{P}(X_{i,j}=1|v_j=v) \\
&= \sum_{j=j'-l_0+1}^{j'}\sum_{v=V_1}^{V_L} h_v p_{i,j|v} \le 1,
\end{aligned}$$

for all $j' = 1, 2, \ldots, N$. Moreover,

$$\sum_{i=1}^{m}\mathbb{P}(X_{i,j}=1, v_j=v) = \sum_{i=1}^{m}\mathbb{P}(X_{i,j}=1|v_j=v)h_v = \sum_{i=1}^{m} p_{i,j|v}h_v \le h_v,$$

for $v = V_1, V_2, \ldots, V_L$.

Let $q_{i,j,v} \equiv p_{i,j|v} h_v$. Then the primal program is

$$\max \frac{1}{N} \sum_{i=1}^{m} \sum_{j=1}^{N} \sum_{v=V_1}^{V_L} w_i v q_{i,j,v}, \tag{P2}$$

$$s.t. \sum_{j=j'-l_0+1}^{j'} \sum_{v=V_1}^{V_L} q_{i,j,v} \leq 1, \qquad \text{for } i = 1, 2, \ldots, m, \text{ and } j' = 1, 2, \ldots, N, \tag{6.31}$$

$$\sum_{i=1}^{m} q_{i,j,v} \leq h_v, \qquad \text{for } j = 1, 2, \ldots, N, \text{ and } v = V_1, V_2, \ldots, V_L. \tag{6.32}$$

The corresponding dual program is

$$\min \sum_{i=1}^{m} \sum_{j=1}^{N} \alpha_{i,j} + \sum_{j=1}^{N} \sum_{v=V_1}^{V_L} \beta_{j,v} h_v, \tag{D2}$$

$$s.t. \sum_{j'=j}^{j+l_0-1} \alpha_{i,j'} + \beta_{j,v} \geq \frac{1}{N} v w_i,$$

$$\text{for } i = 1, 2, \ldots, m, \text{ and } j = 1, 2, \ldots, N, \text{ and } v = V_1, V_2, \ldots, V_L, \tag{6.33}$$

where $\alpha_{i,j}$ and $\beta_{j,v}$ are the variables corresponding to constraints (6.31) and (6.32), respectively. The complementary slackness conditions for the primal

program (P2) and the dual program (D2) are

$$\alpha_{i,j'}\left(\sum_{j=j'-l_0+1}^{j'}\sum_{v=V_1}^{V_L} q_{i,j,v}-1\right)=0,$$

$$\text{for } i=1,2,\ldots,m \text{ and } j'=1,2,\ldots,N, \tag{6.34}$$

$$\beta_{j,v}\left(\sum_{i=1}^{m} q_{i,j,v}-h_v\right)=0,$$

$$\text{for } j=1,2,\ldots,N, \text{ and } v=V_1,V_2,\ldots,V_L, \tag{6.35}$$

$$q_{i,j,v}\left(\sum_{j'=j}^{j+l_0-1}\alpha_{i,j'}+\beta_{j,v}-\frac{vw_i}{N}\right)=0,$$

$$\text{for } i=1,2,\ldots,m \text{ and } j=1,2,\ldots,N,$$
$$\text{and } v=V_1,V_2,\ldots,V_L. \tag{6.36}$$

Theorem 36 provides an upper bound for the optimal expected reward for scheduling IID equal-length jobs on multiple weighted machines. We define the *quantile* $x=F^{-1}(y)$ for a discrete cumulative distribution function $y=F(x)$ as

$$x=F^{-1}(y)=\begin{cases}X_i, & \text{if } y=F(X_i),\\ X_{i-1}, & \text{if } F(X_{i-1})<y<F(X_i),\end{cases}$$

for $i=1,2,\ldots,n$, where $0=X_0<X_1<X_2<\ldots<X_n$ is the discrete support for x.

Theorem 36. *An upper bound for scheduling IID equal-length jobs on multiple weighted machines is*

$$\frac{\tilde{V}_{m-1}}{l_0}\sum_{i=1}^{m} w_i + w_m \sum_{v=\tilde{V}_{m-1}}^{V_L}(v-\tilde{V}_{m-1})h_v, \tag{6.37}$$

where $\tilde{V}_{m-1}\equiv F^{-1}(1-\frac{1}{l_0})$ *is the quantile of the cumulative distribution function of job values.*

Proof:

We construct a solution to the dual program (D2) first, and show that the solution is feasible for the dual program. We then establish an upper bound

for the value of the primal program using the value of the dual program based on this solution.

Set $\tilde{V}_{m-1} = F^{-1}(1 - \frac{1}{l_0})$. We construct a dual solution as follows:

$$\beta_{j,v} = \beta_v = \begin{cases} 0, & \text{for } v < \tilde{V}_{m-1}, \\ \frac{1}{N}(v - \tilde{V}_{m-1})w_m, & \text{for } \tilde{V}_{m-1} \leq v \leq V_L, \end{cases} \tag{6.38}$$

for $j = 1, 2, \ldots, N$ and $v \in \{V_1, V_2, \ldots, V_L\}$. The values of $\alpha_{i,j}$ are

$$\alpha_{i,j} = \alpha_i = \frac{1}{N} \frac{w_i \tilde{V}_{m-1}}{l_0}, \tag{6.39}$$

for $j = 1, 2, \ldots, N$ and $i = 1, 2, \ldots, m$.

To see that the dual solution given by (6.38) and (6.39) is a feasible solution to (D2), we need to show that constraint (6.33) is satisfied. For $\tilde{V}_{m-1} \leq v \leq V_L$ and $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, N$,

$$\begin{aligned} \sum_{j'=j}^{j+l_0-1} \alpha_{i,j'} + \beta_{j,v} &= l\alpha_i + \beta_v \\ &= \frac{1}{N} w_i \tilde{V}_{m-1} + \frac{1}{N}(v - \tilde{V}_{m-1})w_m \\ &\geq \frac{1}{N} w_i v, \end{aligned}$$

where the first two equalities follow by substituting the variables of $\alpha_{i,j}$ and $\beta_{j,v}$ with the values given by (6.39) and (6.38); the last inequality can be verified by rearranging terms on both sides. For $v < \tilde{V}_{m-1}$ and $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, N$,

$$\sum_{j'=j}^{j+l_0-1} \alpha_{i,j'} + \beta_{j,v} = l\alpha_i = \frac{1}{N} w_i \tilde{V}_{m-1} \geq \frac{1}{N} w_i v.$$

Therefore, the dual solution given by (6.39) and (6.38) is a feasible solution to (D2).

From weak duality, an upper bound for the value of the primal program is the value of the dual program based on the solution given by (6.38) and

(6.39), and hence

$$\sum_{i=1}^{m}\sum_{j=1}^{N}\frac{1}{N}\frac{w_i\tilde{V}_{m-1}}{l_0}+\sum_{j=1}^{N}\sum_{v=\tilde{V}_{m-1}}^{V_L}\frac{1}{N}(v-\tilde{V}_{m-1})w_m h_v$$
$$=\frac{\tilde{V}_{m-1}}{l_0}\sum_{i=1}^{m}w_i+w_m\sum_{v=\tilde{V}_{m-1}}^{V_L}(v-\tilde{V}_{m-1})h_v.$$

■

6.3 Approximation Algorithms for Adversarial Online Interval Scheduling Problems

This section uses the primal-dual technique to analyze approximation algorithms for scheduling adversarial C-benevolent job sequences on a single machine. To simplify notations, we eliminate the subscript i in all our variables and assume $w_1 = 1$. Note that for adversarial job sequences, it is not guaranteed that one job arrives at the beginning of each time slot. We formulate the scheduling problem for C-benevolent jobs on a single machine as a primal program as follows, with the corresponding dual program:

Primal program:

$$\max\sum_{j=1}^{N}X_j v_j, \tag{P3}$$
$$s.t.\sum_{j:t\in[a_j,f_j)}X_j\le 1, \qquad \text{for } t=1,2,\ldots,T, \tag{6.40}$$
$$X_j\le 1, \qquad \text{for } j=1,2,\ldots,N, \tag{6.41}$$

where constraint (6.41) is a linear relaxation for binary assignment variables $\{X_j\in\{0,1\}\}$.

Dual program:

$$\min \sum_{t=1}^{T} u_t + \sum_{j=1}^{N} s_j, \tag{D3}$$

$$s.t. \quad \sum_{t:t\in[a_j,f_j)} u_t + s_j \geq v_j, \qquad \text{for } j = 1, 2, \ldots, N, \tag{6.42}$$

$$u_t \geq 0, \qquad \text{for } t = 1, 2, \ldots, T, \tag{6.43}$$

$$s_j \geq 0, \qquad \text{for } j = 1, 2, \ldots, N, \tag{6.44}$$

where $\{u_t\}$ and $\{s_j\}$ correspond to constraints (6.40) and (6.41), respectively. The dual variables $\{u_t\}$ and $\{s_j\}$ denote the *basic cost* for each time slot t and the *additional cost* for the j^{th} arrived job, respectively.

The construction of a feasible solution to the dual program (D3) depends on the specific job instance, and hence there is no general dual solution that is feasible for all possible job instance. We turn to constructing a dual solution that is feasible for the subset of jobs that are completed by the optimal schedule for a job instance $\mathbf{I}$, denoted by $\mathbf{I}_{opt} \subset \mathbf{I}$, and show that weak duality still holds. Specifically, we consider the following *restricted dual program*:

Restricted Dual program:

$$\min \sum_{t=1}^{T} u_t + \sum_{j=1}^{N} s_j, \tag{D4}$$

$$s.t. \quad s_j + \sum_{t:t\in[a_j,f_j)} u_t \geq v_j, \qquad \text{for } J_j \in \mathbf{I}_{opt}, \tag{6.45}$$

$$u_t \geq 0, \qquad \text{for } t = 1, 2, \ldots, T, \tag{6.46}$$

$$s_j \geq 0, \qquad \text{for } J_j \in \mathbf{I}_{opt}. \tag{6.47}$$

Proposition 8 gives a modified weak duality, which reduces the problem of constructing a feasible solution to the dual program (D3) to constructing a feasible solution to the restricted dual program (D4) without modifying the upper bound for the optimal reward.

Proposition 8.

$$OPT(\mathbf{I}) \leq D4(\mathbf{I}_{opt}),$$

for all job instances $\mathbf{I}$*, where* $OPT(\mathbf{I})$ *denotes the reward for the optimal*

schedule for job instance $\mathbf{I}$ *and* $D4(\mathbf{I}_{opt})$ *denotes the optimal value of the restricted dual program* (D4) *for job instance* $\mathbf{I}_{opt}$.

Proof: Since $\mathbf{I}_{opt}$ denotes the subset of jobs that are assigned by the optimal algorithm, then

$$OPT(\mathbf{I}) = OPT(\mathbf{I}_{opt}) \leq D4(\mathbf{I}_{opt}),$$

where $OPT(\mathbf{I})$ and $OPT(\mathbf{I}_{opt})$ denote the reward for the optimal schedule for jobs in $\mathbf{I}$ and $\mathbf{I}_{opt}$, respectively, and the inequality follows from weak duality. ■

Note that $\mathbf{I}_{opt}$ depends on the specific job instance $\mathbf{I}$. Since the competitive ratio of an algorithm considers the worst-case performance, we characterize the worst-case $\mathbf{I}_{opt}$, which has the largest reward, for a job instance $\mathbf{I}$ for each algorithm, and then apply primal-dual technique on this specific instance, which is sufficient for computing the competitive ratio of an algorithm.

6.3.1 Deterministic algorithm

This section considers a deterministic *Greedy-α* algorithm: whenever a new job J_{new} arrives, if the machine is idle, then assign job J_{new}; otherwise, the machine must be executing some job J_{cur}, in which case terminate J_{cur} and assign J_{new} if and only if $v(J_{new}) > \alpha v(J_{cur})$, where $v(J)$ denotes the value of job J and $\alpha \geq 1$ is the *abortion ratio*. We use the primal-dual techniques to compute the competitive ratio of the Greedy-α algorithm and show that when $\alpha = 2$, the Greedy-α algorithm has the smallest competitive ratio of 4, which is consistent with [39].

We first clarify some definitions needed for the analysis. Consider a job J completed under the Greedy-α algorithm on a single machine. Then all jobs that are previously assigned by the Greedy-α algorithm but later aborted in favor of J are labelled *predecessors* of J. The job that has the largest completion time among jobs arriving during the execution of J but not assigned by the Greedy-α algorithm is labelled the *successor* of J. The subset of jobs consisting of all predecessors of J, job J, and the successor of J is referred to as the *segment* of J (see Figure 6.1). Then, from *Observation 3.1* in [39], a job sequence can be divided into non-overlapping segments of all completed jobs under the Greedy-α algorithm (i.e., no jobs arrive during the gap between subsequent segments, if such a gap exists). We compute the

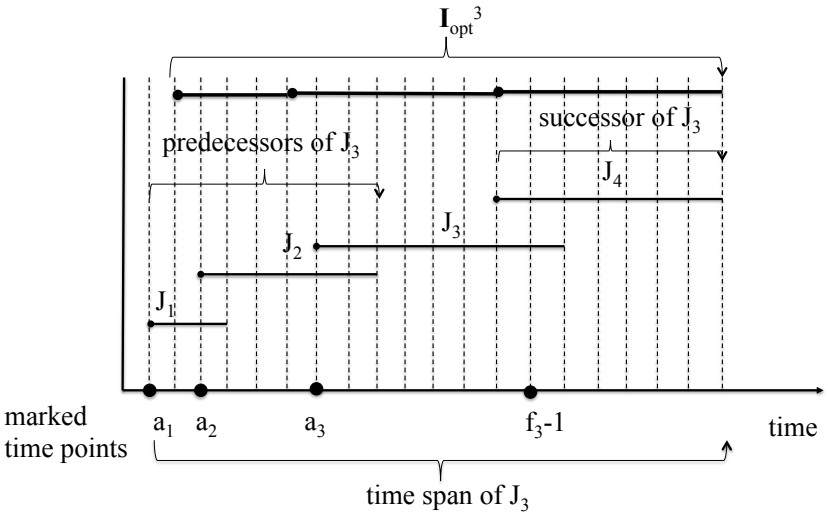

Figure 6.1: An example for a segment of J_k, with $k = 3$. $\{J_1, J_2\}$ are the predecessors of J_3, and J_4 is the successor of J_3. $\{a_1, a_2, a_3, f_3 - 1\}$ are the marked time points of the segment of J_3. $[a_1, f_4)$ is the time span of the segment of J_3. $\mathbf{I}^3_{opt}$ is the set of jobs in the optimal schedule covered by the span of the segment of J_3.

competitive ratio of the Greedy-α algorithm for any segment, which is the same as the competitive ratio of the Greedy-α algorithm for the whole job sequence.

Let $\{J_j\}_{j=1}^{k+1}$ denote the segment of J_k, where J_k is the job completed by the Greedy-α algorithm, $\{J_j\}_{j=1}^{k-1}$ is the set of predecessors of J_k, and J_{k+1} is the successor of J_k. Then the time interval, which starts from the arrival time of J_1 (closed) and ends at the completion time of J_{k+1} (open), is referred to as the *time span* of the segment of J_k. The time points $a_1 < a_2 \ldots < a_k < f_k - 1$ are referred to as the *marked time points* of the segment of J_k, where $\{a_j\}_{j=1}^{k}$ denote the arrival times of $\{J_j\}_{j=1}^{k}$ and f_k denote the completion times of J_k.

Let $\mathbf{I}^k_{opt}$ denote the set of jobs in the optimal schedule covered by the span of the segment of J_k (i.e., for any job in $\mathbf{I}^k_{opt}$, its interval is within the time span of the segment of J_k, see Figure 6.1). Assumption 1 characterizes the worst-case $\mathbf{I}^k_{opt}$ for a job instance for the Greedy-α algorithm.

Assumption 1. *(a) Any job $\tilde{J} \in \mathbf{I}^k_{opt}$ contains at least one marked time point (except the first job).*
(b) The union of job intervals in $\mathbf{I}^k_{opt}$ covers the entire time interval of $[a_1 + 1, f_{k+1})$.

This assumption may only increase the reward for the optimal schedule due to the convexity of C-benevolent value-length function [39, 44]: if there is some job in the optimal schedule (except the first job) that does not contain any marked time point, we can change lengths of some jobs in the optimal schedule such that resulting jobs all contain at least one marked time point and have a larger reward.

From Proposition 8, we construct a feasible solution to the restricted dual program (D4) to obtain an upper bound for the optimal reward. We start by setting all the dual variables to zero. Therefore, $u_t = 0$ for all t and $s_j = 0$ for all j. Moreover, we keep $s_j = 0$ for all j unchanged throughout the scheduling of the whole job instance. We increase the value of the corresponding u_t when a job is assigned by the Greedy-α algorithm (i.e., when the value of (P3) increases). Note that the assignment of J_1 results in an increase of $v(J_1)$ for the value of (P3). When J_2 is assigned and J_1 is aborted, the assignment of J_2 results in an increase of $\Delta v_2 = v(J_2) - v(J_1)$ for the value of (P3). Similarly, every subsequent job J_j in the segment will result in an increase of $\Delta v_j = v(J_j) - v(J_{j-1})$ for the value of (P3), for $j = 2, 3, \ldots, k$. We construct the dual solution as follows:

$$\begin{aligned} u_{a_1+1} &= \alpha^{-1} v(J_1) \times \gamma, \\ u_{a_j} &= \Delta v_{j-1} \times \gamma, \qquad\qquad \text{for } j = 2, 3, \ldots, k, \\ u_{f_k-1} &= \Delta v_k \times \gamma, \end{aligned} \tag{6.48}$$

where $\Delta v_j = v(J_j) - v(J_{j-1})$ with $v(J_0) \equiv \alpha^{-1} v(J_1)$ and $\gamma > 1$ is the competitive ratio of the algorithm to be determined later.

We are left to show that the dual solution given by (6.48) is feasible for the restricted dual program (D4). Consider any job $\tilde{J}_z \in \mathbf{I}_{opt}$ whose interval satisfies $[a_z, f_z) \subset [a_1 + 1, f_{k+1})$. We consider two cases: (a) $\tilde{J}_z \in \{J_1, J_2, \ldots, J_k\}$, and (b) $\tilde{J}_z \notin \{J_1, J_2, \ldots, J_k\}$. Consider case (a) first.
For $\tilde{J}_z = J_1$,

$$\begin{aligned} \sum_{t: t \in [a_z, f_z)} u_t &= u_{a_1+1} + u_{a_2} \\ &= \alpha^{-1} v(J_1) \times \gamma + \Delta v_1 \times \gamma \\ &= v(J_1) \times \gamma. \end{aligned} \tag{6.49}$$

For $\tilde{J}_z = J_j$, $j = 2, 3, \ldots, k-1$,

$$\begin{aligned}\sum_{t:t\in[a_z,f_z)} u_t &= u_{a_j} + u_{a_{j+1}} \\ &= (\Delta v_{j-1} + \Delta v_j) \times \gamma \\ &= (v(J_j) - v(J_{j-2})) \times \gamma \\ &\geq (1-\alpha^{-2}) v(J_j) \times \gamma. \end{aligned} \tag{6.50}$$

For $\tilde{J}_z = J_k$,

$$\begin{aligned}\sum_{t:t\in[a_z,f_z)} u_t &= u_{a_k} + u_{f_k-1} \\ &= (\Delta v_{k-1} + \Delta v_k) \times \gamma \\ &= (v(J_k) - v(J_{k-2})) \times \gamma \\ &\geq (1-\alpha^{-2}) v(J_k) \times \gamma, \end{aligned} \tag{6.51}$$

where inequalities (6.50) and (6.51) follow from the abortion rule of the Greedy-α algorithm: $v(J_j) > \alpha v(J_{j-1})$, for $j = 2, 3, \ldots, k$. Therefore, for (6.49) to (6.51) to satisfy constraint (6.45) of the restricted dual program requires

$$\min\{\gamma, (1-\alpha^{-2})\gamma\} \geq 1 \qquad \Leftrightarrow \qquad \gamma \geq \frac{1}{1-\alpha^{-2}}. \tag{6.52}$$

For case (b), if $\tilde{J}_z$ is the first job in $\mathbf{I}^k_{opt}$ and does not contain any marked time point, then $f_z < a_2 < f_1$, and hence $v(\tilde{J}_z) < v(J_1)$. From Assumption 1, since $\mathbf{I}^k_{opt}$ covers the entire time interval $[a_1+1, f_{k+1})$, $\tilde{J}_z$ must contain a_1+1 (i.e., $a_1+1 \in [a_z, f_z)$); therefore,

$$\sum_{t:t\in[a_z,f_z)} u_t = u_{a_1+1} = \alpha^{-1} v(J_1) \times \gamma \geq \alpha^{-1} v(\tilde{J}_z) \times \gamma. \tag{6.53}$$

Otherwise, there exists some marked time point contained in $[a_z, f_z)$, the interval of $\tilde{J}_z$. Since job $\tilde{J}_z$ is not assigned by the Greedy-α algorithm, then $v(\tilde{J}_z) \leq \alpha v(J_{k_z})$ for some $k_z \in \{1, 2, \ldots, k\}$, where J_{k_z} is the job assigned by the Greedy-α algorithm when job $\tilde{J}_z$ arrives.

For $k_z = 1, 2, \ldots, k-1$,

$$\begin{aligned}
\sum_{t:t\in[a_z,f_z)} u_t &\geq u_{a_{k_z+1}} \\
&= \Delta v_{k_z} \times \gamma \\
&= (v(J_{k_z}) - v(J_{k_z-1})) \times \gamma \\
&\geq (1-\alpha^{-1}) v(J_{k_z}) \times \gamma \qquad (6.54) \\
&\geq (1-\alpha^{-1})\alpha^{-1} v(\tilde{J}_z) \times \gamma. \qquad (6.55)
\end{aligned}$$

For $k_z = k$,

$$\begin{aligned}
\sum_{t:t\in[a_z,f_z)} u_t &\geq u_{f_k-1} \\
&= \Delta v_k \times \gamma \\
&= (v(J_k) - v(J_{k-1})) \times \gamma \\
&\geq (1-\alpha^{-1}) v(J_k) \times \gamma \qquad (6.56) \\
&\geq (1-\alpha^{-1})\alpha^{-1} v(\tilde{J}_z) \times \gamma, \qquad (6.57)
\end{aligned}$$

where inequalities (6.54) and (6.56) follow from the abortion rule of the Greedy-α algorithm: $v(J_j) > \alpha v(J_{j-1})$, for $j = 2, 3, \ldots, k$. Therefore, for (6.53) to (6.57) to satisfy constraint (6.45) of the restricted dual program requires

$$\min\{\alpha^{-1}\gamma, (1-\alpha^{-1})\alpha^{-1}\gamma\} \geq 1 \qquad \Leftrightarrow \qquad \gamma \geq \frac{\alpha}{1-\alpha^{-1}}. \qquad (6.58)$$

Since $\frac{\alpha}{1-\alpha^{-1}} \geq \frac{1}{1-\alpha^{-2}}$, then the lower bound for γ given by (6.58) dominates the lower bound given by (6.52). Minimizing the lower bound for γ given by (6.58) over the value of α gives the optimal value for the abortion ratio as $\alpha^* = 2$, and the competitive ratio for the Greedy-2 algorithm is $\gamma^* = 4$.

6.3.2 Randomized algorithm

This section considers a randomized Greedy algorithm, BIT, proposed by [44] for scheduling C-benevolent jobs on a single machine. With probability $0 < p < 1$, the algorithm assigns every job according to the Greedy-α algorithm. We divide the whole job sequence into non-overlapping segments of completed

jobs by the Greedy-α algorithm, as described in Section 6.3.1. Consider the segment of J_k, $\{J_1, J_2, \ldots, J_{k+1}\}$, where job J_k is the only completed job under the Greedy-α algorithm. With probability $1-p$, the algorithm assigns every other job in a segment according to the Greedy-α algorithm. For example, the algorithm assigns jobs with odd arrival orders in the segment of J_k, $\{J_1, J_3, \ldots, J_{2\lfloor(k-1)/2\rfloor+1}\}$, with probability $(1-p)/2$ and jobs with even arrival orders in the segment of J_k, $\{J_2, J_4, \ldots, J_{2\lfloor k/2\rfloor}\}$, with probability $(1-p)/2$. We refer to this algorithm as the p-Greedy-α algorithm. [44] proves a competitive ratio of $2+\sqrt{3}$ when $\alpha = 1+1/\sqrt{3}$ and $p = 1/\sqrt{3}$ using analysis techniques similar to [39]. We provide a matching competitive ratio using the primal-dual technique.

When the p-Greedy-α algorithm is assigning every job in a segment, we say the algorithm is in *normal mode*; when the algorithm is assigning every job with odd (even) arrival orders in a segment, we say the algorithm is in *random odd (even) mode*. For the segment of J_k, $\{J_1, J_2, \ldots, J_{k+1}\}$, let $\{a_1, a_2, \ldots, a_k\}$ denote the arrival times of jobs $\{J_1, J_2, \ldots, J_k\}$ and f_k denote the completion time of jobs J_k. Then $\{a_1, a_2, \ldots, a_k, f_k - 1\}$ is the set of the marked times points of the segment of J_k. Note that the p-Greedy-α algorithm can complete job J_k with at least probability p.

Since a job sequence can be divided into non-overlapping segments of completed jobs, we compute the competitive ratio of the p-Greedy-α algorithm for any segment, which is the same as the competitive ratio of the p-Greedy-α algorithm for the entire job sequence. Note that we only need to consider the worst-case job instance to compute the competitive ratio. The worst-case job instance for the p-Greedy-α algorithm is given by *Lemma 3.5* [44], which is rephrased in Lemma 9.

Lemma 9. *When the competitive ratio γ and the parameters of the p-Greedy-α algorithm satisfy*

$$\gamma(1-p) \leq \alpha, \tag{6.59}$$

the worst-case job instance for the p-Greedy-α algorithm satisfies $f_{j-2} \leq a_j$, for $j = 3, 4, \ldots, k+1$ and $k \geq 3$, where $\{a_1, a_2, \ldots, a_{k+1}\}$ and $\{f_1, f_2, \ldots, f_{k+1}\}$ are the arrival and completion times of jobs in a segment, $\{J_1, J_2, \ldots, J_{k+1}\}$.

We now consider the segment of J_k in the worst-case job instance given by Lemma 9 and construct a solution to the restricted dual program (D4). Let $\mathbf{I}^k_{opt}$ denote the set of jobs in the optimal schedule covered by the span

of the segment of J_k. We make Assumption 1 for $\mathbf{I}_{opt}^k$, since this assumption only increases the reward for the optimal schedule due to the convexity of C-benevolent jobs.

We initialize the dual variables as $u_t = 0$ for all t and $s_j = 0$ for all j. The values of $\{s_j\}$ remain zero throughout the scheduling of the entire job sequence. Each time an assignment is made (either some previously assigned job is aborted or not), we increase the value of the corresponding u_t. Since the algorithm has three different modes, we describe the rules for increasing the values of u_t separately. In the normal mode, the values of u_t are set in the same way as for the deterministic Greedy-α algorithm (see (6.48)). In the random odd (even) mode, only jobs with odd (even) arrival orders in a segment can be assigned or aborted. For the worst-case job instance given by Lemma 9, $f_{j-2} \leq a_j$ for $j = 3, 4, \ldots, k+1$, and hence jobs $\{J_1, J_3, \ldots, J_{2\lfloor (k-1)/2 \rfloor + 1}\}$ ($\{J_2, J_4, \ldots, J_{2\lfloor k/2 \rfloor}\}$) are completed in the random odd (even) mode. In the random odd mode, for $1 \leq j \leq k$ and $mod(j, 2) = 1$, set u_t as follows:

$$u_{f_j - 1} = v(J_j) \times \gamma. \tag{6.60}$$

In the random even mode, for $1 \leq j \leq k$ and $mod(j, 2) = 0$, set u_t as follows:

$$u_{f_j - 1} = v(J_j) \times \gamma. \tag{6.61}$$

For the dual solution given by (6.48), (6.60) and (6.61), the increase in the value of the primal program (P3) is no less than $1/\gamma$ of the increase in the value of the dual program (D4). Therefore, we are left to show that the constructed dual solution is feasible for the restricted dual program (D4) to prove that the competitive ratio of the p-Greedy-α algorithm is γ.

We consider two cases for a job $\tilde{J}_z$ in the optimal schedule $\mathbf{I}_{opt}^k$: (a) job $\tilde{J}_z \in \{J_1, J_2, \ldots, J_k\}$ and hence is assigned in the normal mode (either later aborted or completed); (b) job $\tilde{J}_z \notin \{J_1, J_2, \ldots, J_k\}$ and hence is not assigned in the normal mode.

Consider case (a). Then $\tilde{J}_z = J_j$, for some $1 \leq j \leq k$. If $j = 1$, then

$$\mathbb{E}[\sum_{t: t \in [a_1, f_1)} u_t] \geq \mathbb{E}[u_{a_1+1} + u_{f_1 - 1}] = pv(J_1)\gamma + \frac{1-p}{2} v(J_1)\gamma = \frac{1+p}{2}\gamma v(J_1). \tag{6.62}$$

If $2 \leq j \leq k$, then

$$\begin{aligned}
&\mathbb{E}[\sum_{t:t\in[a_j,f_j)} u_t] \\
\geq&\mathbb{E}[u_{a_j} + u_{a_{j+1}} + u_{f_j-1}] \\
\geq&p(v(J_j) - v(J_{j-1}))\gamma + p(v(J_{j-1}) - v(J_{j-2}))\gamma + \frac{1-p}{2}(v(J_j) + v(J_{j-1}))\gamma \\
\geq&\left(p(1-\alpha^{-2}) + \frac{1-p}{2}(1+\alpha^{-1})\right)\gamma v(J_j). \qquad (6.63)
\end{aligned}$$

For (6.62) and (6.63) to satisfy constraint (6.45) of the restricted dual program requires

$$\min\{\frac{1+p}{2}, p(1-\alpha^{-2}) + \frac{1-p}{2}(1+\alpha^{-1})\}\gamma \geq 1. \qquad (6.64)$$

Next we consider case (b). If $\tilde{J}_z$ is the first job in $\mathbf{I}^k_{opt}$ and does not contain any marked time point, then $f_z < a_2 < f_1$, and hence, $v(\tilde{J}_z) < v(J_1)$. From Assumption 1, since $\mathbf{I}^k_{opt}$ covers the entire time interval $[a_1+1, f_{k+1})$, $\tilde{J}_z$ must contain $a_1 + 1$ (i.e., $a_1 + 1 \in [a_z, f_z)$); therefore,

$$\mathbb{E}[\sum_{t:t\in[a_z,f_z)} u_t] \geq \mathbb{E}[u_{a_1+1}] = p\alpha^{-1}\gamma v(J_1) \geq p\alpha^{-1}\gamma v(\tilde{J}_z). \qquad (6.65)$$

Otherwise, there exists a marked time point contained in $[a_z, f_z)$, the interval of job $\tilde{J}_z$. Therefore, if the marked time point is a_{k_z} for $k_z = 2, 3, \ldots, k$, then

$$\begin{aligned}
\mathbb{E}[\sum_{t:t\in[a_z,f_z)} u_t] &\geq \mathbb{E}[u_{a_{k_z}} + u_{f_{k_z-1}-1}] \\
&= p(v(J_{k_z-1}) - v(J_{k_z-2}))\gamma + \frac{1-p}{2}v(J_{k_z-1})\gamma \\
&\geq \left(p(1-\alpha^{-1}) + \frac{1-p}{2}\right)\gamma v(J_{k_z-1}), \qquad (6.66)
\end{aligned}$$

where $v(\tilde{J}_z) \leq \alpha v(J_{k_z-1})$. If the marked time point is $f_k - 1$, then

$$\begin{aligned}\mathbb{E}[\sum_{t:t\in[a_z,f_z)} u_t] &\geq \mathbb{E}[u_{f_k-1}] \\ &= p(v(J_k) - v(J_{k-1}))\gamma + \frac{1-p}{2}v(J_k)\gamma \\ &\geq \left(p(1-\alpha^{-1}) + \frac{1-p}{2}\right)\gamma v(J_k), \end{aligned} \tag{6.67}$$

where $v(\tilde{J}_z) < \alpha v(J_k)$.

For (6.65) to (6.67) to satisfy constraint (6.45) requires

$$\begin{aligned}\frac{p}{\alpha}\gamma &\geq 1, \\ \left(p(1-\alpha^{-1}) + \frac{1-p}{2}\right)\gamma &\geq \alpha. \end{aligned} \tag{6.68}$$

Since

$$\alpha^{-1}\left(p(1-\alpha^{-1}) + \frac{1-p}{2}\right) \leq p(1-\alpha^{-2}) + \frac{1-p}{2}(1+\alpha^{-1}),$$

and

$$\alpha^{-1}\left(p(1-\alpha^{-1}) + \frac{1-p}{2}\right) = \alpha^{-1}\left(\frac{1+p}{2} - \frac{p}{\alpha}\right) \leq \frac{1+p}{2},$$

for $\alpha \geq 1$, then combining the conditions given by (6.59), (6.64) and (6.68) leads to

$$\max\{\frac{\alpha}{p}, (\frac{1+p}{2} - \frac{p}{\alpha})^{-1}\alpha\} \leq \gamma \leq \frac{\alpha}{1-p}. \tag{6.69}$$

Substituting the values of parameters p and α with the values $\alpha = 1 + 1/\sqrt{3}$ and $p = 1/\sqrt{3}$, the competitive ratio of the p-Greedy-α algorithm can be computed as $\gamma = 2 + \sqrt{3}$ from (6.69), which matches the competitive ratio given by [44].

6.3.3 Cooperative Greedy algorithm

This section considers a randomized Cooperative Greedy algorithm for scheduling C-benevolent jobs on a single machine. The Cooperative Greedy algorithm was originally proposed by [45] for scheduling C-benevolent jobs on a single machine and [70] for scheduling C-benevolent jobs on two machines.

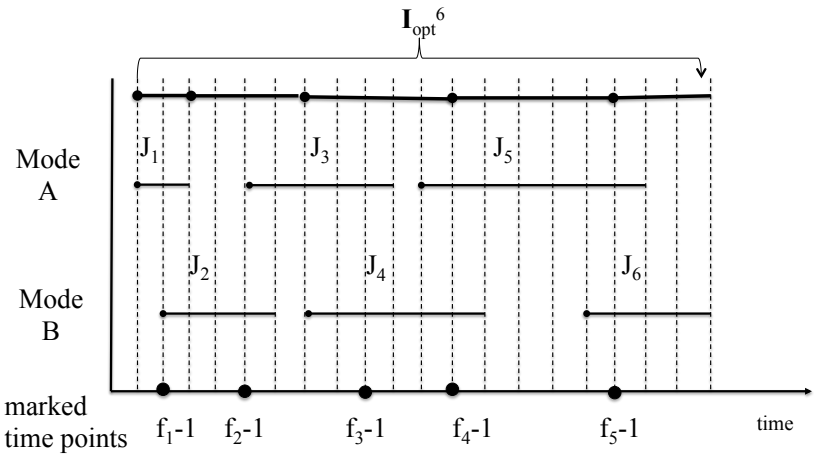

Figure 6.2: An example for a segment $\{J_1, J_2, J_3, J_4, J_5, J_6\}$ for the Cooperative Greedy algorithm. $\{f_1 - 1, f_2 - 1, f_3 - 1, f_4 - 1, f_5 - 1\}$ are the marked time points of the segment. $\mathbf{I}_{opt}^6$ is the set of jobs in the optimal schedule covered by the span of the segment, $[a_1, f_6)$.

The Cooperative Greedy algorithm initially chooses one of two modes, A and B, with equal probability and sticks to that mode thereafter. Let J_1 denote the first arriving job when the machine is available. Mode A assigns and completes job J_1; mode B does not assign job J_1 and uses the Greedy-1 algorithm for jobs arriving during $[a_1+1, f_1)$. At time f_1, if mode B schedules no job on the machine, we say a *segment* (i.e., $\{J_1\}$) ends. Otherwise, let J_2 denote the job scheduled on the machine by mode B at time f_1. Then during $[f_1, f_2)$, mode A uses the Greedy-1 algorithm to schedule jobs; mode B completes job J_2. At time f_2, if mode A schedules no job on the machine, we say a segment (i.e.,$\{J_1, J_2\}$) ends. Otherwise, let J_3 denote the job scheduled on the machine by mode A at time f_2. Then during $[f_2, f_3)$, mode A completes job J_3; mode B uses the Greedy-1 algorithm to schedule jobs. This process continues until no job is scheduled on the machine in either mode A or B, and we say a segment ends (see Figure 6.2). When the next job arrives, a new segment starts and the algorithm continues this process until the end of a job instance.

Let $\{J_1, J_2, \ldots, J_k\}$ denote a segment with k completed jobs in either mode under the Cooperative Greedy algorithm. Then each job J_j is completed with probability 1/2, for $j = 1, 2, \ldots, k$. Time points $\{f_1 - 1, f_2 -$

$1, \ldots, f_{k-1} - 1\}$ are defined as the marked time points for the the segment, where $\{f_1, f_2, \ldots, f_{k-1}\}$ denote the completion times of jobs $\{J_1, J_2, \ldots, J_{k-1}\}$. Since a job sequence can be divided into non-overlapping segments, we compute the competitive ratio of the Cooperative Greedy algorithm for any segment, which is the same as the competitive ratio of the Cooperative Greedy algorithm for the entire job sequence.

We first characterize the worst-case job instance for the Cooperative Greedy algorithm, which is sufficient to consider for computing the competitive ratio. Lemma 10 gives a criterion to simplify jobs in the optimal schedule without reducing the total reward.

Lemma 10. *For any segment $\{J_1, J_2, \ldots, J_k\}$ ($k \geq 2$) in a job instance* **I**, *the reward of any feasible schedule covered by the time interval $[a_1, f_k)$ can be increased by reallocating job lengths if there is some job (except the first job) in the schedule that does not contain any marked time point, where a_1 is the arrival time of J_1 and f_k is the completion time of J_k.*

Proof:

The proof is by induction on k.

Consider the base case of $k = 2$. Let $\{\tilde{J}_j\}_{j=1}^{h}$ denote a feasible schedule covered by the time span $[a_1, f_2)$ of segment $\{J_1, J_2\}$, where $\tilde{J}_j$ is reordered increasingly with respect to the lengths (i.e., $a(\tilde{J}_j) < a(\tilde{J}_{j+1})$ and $l(\tilde{J}_j) < l(\tilde{J}_{j+1})$, for $j = 1, 2, , \ldots, h-1$). Then,

$$\sum_{j=1}^{h} l(\tilde{J}_j) \leq f(\tilde{J}_h) - a(\tilde{J}_1) \leq f_2 - a_1, \tag{6.70}$$

where $a(J)$ and $f(J)$ denote the arrival and completion times of job J. If $\tilde{J}_h$ does not contain the marked time point $f_1 - 1$, then we construct two new jobs $\bar{J}_1$ and $\bar{J}_2$, with $a(\bar{J}_1) = a(\tilde{J}_1)$, $l(\bar{J}_1) = a_2 - a(\tilde{J}_1)$ and $a(\bar{J}_2) = a_2$, $l(\bar{J}_2) = f(\tilde{J}_h) - a_2$, where a_2 is the arrival time of job J_2 and $l(J)$ denotes the length of job J. Then $\{\bar{J}_1, \bar{J}_2\}$ is a new feasible schedule covered by $[a_1, f_2)$. Moreover, $l(\bar{J}_1) + l(\bar{J}_2) = f(\tilde{J}_h) - a(\tilde{J}_1) \geq \sum_{j=1}^{h} l(\tilde{J}_j)$ and $l(\bar{J}_2) > l(\tilde{J}_h)$. Therefore, from the property of C-benevolent jobs,

$$v(\bar{J}_1) + v(\bar{J}_2) \geq \sum_{j=1}^{h} v(\tilde{J}_j).$$

Assume that Lemma 10 holds for $k = k_0 - 1$. To prove that it holds for $k = k_0$, let $\{\tilde{J}_j\}_{j=1}^{h'}$ denote a feasible schedule for the time span $[a_1, f_{k_0})$ of segment $\{J_1, J_2, \ldots, J_{k_0}\}$, where $\tilde{J}_j$ is reordered increasingly with respect to their lengths. If $\tilde{J}_{h'}$ contains the marked time point $f_{k_0-1} - 1$, $f(\tilde{J}_{h'-1}) \leq a(\tilde{J}_{h'}) < f_{k_0-1}$, and hence $\{\tilde{J}_j\}_{j=1}^{h'-1}$ is a feasible schedule covered by interval $[a_1, f_{k_0-1})$. Then the case $k = k_0$ follows from induction assumption for case $k = k_0 - 1$. Otherwise, if $\tilde{J}_{h'}$ does not contain the marked time point $f_{k_0-1} - 1$, let $\tilde{J}_{h_0}$ denote the last job in the feasible schedule whose arrival time is before the arrival time of J_{k_0} in the segment. We then construct two new jobs, $\bar{J}_1$ and $\bar{J}_2$, with $a(\bar{J}_1) = a(\tilde{J}_{h_0})$, $l(\bar{J}_1) = a_{k_0} - a(\tilde{J}_{h_0})$ and $a(\bar{J}_2) = a_{k_0}$, $l(\bar{J}_2) = f(\tilde{J}_{h'}) - a_{k_0}$, where a_{k_0} is the arrival time of job J_{k_0}. Then $\{\tilde{J}_1, \tilde{J}_2, \ldots, \tilde{J}_{h_0-1}, \bar{J}_1, \bar{J}_2\}$ is a new feasible schedule covered by $[a_1, f_{k_0})$. Moreover, since $l(\bar{J}_1) + l(\bar{J}_2) = f(\tilde{J}_{h'}) - a(\tilde{J}_{h_0}) \geq \sum_{j=h_0}^{h'} l(\tilde{J}_j)$ and $l(\bar{J}_2) \geq \max_{h_0+1 \leq j \leq h'} l(\tilde{J}_j)$,

$$v(\bar{J}_1) + v(\bar{J}_2) \geq \sum_{j=h_0}^{h'} v(\tilde{J}_j),$$

which follows from the convexity of C-benevolent jobs. Note that $f(\bar{J}_1) = a(\bar{J}_1) + l(\bar{J}_1) = a_{k_0} \leq f_{k_0-1}$. Therefore, $\{\tilde{J}_1, \tilde{J}_2, \ldots, \tilde{J}_{h_0-1}, \bar{J}_1\}$ is a feasible schedule for the time span $[a_1, f_{k_0-1})$, and hence the case $k = k_0$ follows from the induction assumption for case $k = k_0$, which completes the proof. ■

Let $\mathbf{I}_{opt}^k$ denote the set of jobs in the optimal schedule covered by the time span of segment $[a_1, f_k)$. From Lemma 10, we make Assumption 1 for $\mathbf{I}_{opt}^k$, which only increases the reward for the optimal schedule.

We construct a feasible solution to the restricted dual program (D4) by initializing $u_t = 0$ and $s_j = 0$ for all t and all j. As the Cooperative Greedy algorithm schedules jobs, we increase the values of the corresponding u_t and leave the values of $\{s_j\}$ unchanged. More specifically,

$$\begin{aligned} u_{a_1+1} &= v(J_1) \times \gamma, \\ u_{f_j-1} &= v(J_{j+1}) \times \gamma, \text{ for } j = 1, 2, \ldots, k. \end{aligned} \tag{6.71}$$

We will now show that the dual solution given by (6.71) is feasible for the restricted dual program for the worst-case job instance satisfying Assumption 1. We consider two cases for job $\tilde{J}_z \in \mathbf{I}_{opt}$: (a) job $\tilde{J}_z \in \mathbf{I}_{opt}^k$ is the first job in $\mathbf{I}_{opt}^k$; (b) job $\tilde{J}_z \in \mathbf{I}_{opt}$ is not the first job in $\mathbf{I}_{opt}^k$. For case (a),

by Assumption 1, $a_1 + 1 \in [a_z, f_z)$. If the marked time point $f_1 - 1$ is not contained in $[a_z, f_z)$, then

$$\mathbb{E}[\sum_{t:t\in[a_z,f_z)} u_t] \geq \frac{1}{2}u_{a_1+1} \geq \frac{1}{2}v(J_1) \times \gamma, \tag{6.72}$$

where $v(J_1) \geq v(\tilde{J}_z)$ since $l(J_1) \geq l(\tilde{J}_z)$. If the marked time point $f_1 - 1 \in [a_z, f_z)$, then

$$\mathbb{E}[\sum_{t:t\in[a_z,f_z)} u_t] \geq \frac{1}{2}u_{a_1+1} + \frac{1}{2}u_{f_1-1} \geq \frac{1}{2}(v(J_1) + v(J_2)) \times \gamma, \tag{6.73}$$

where $v(\tilde{J}_z) \leq v(J_2)$.

For case (b), let $f_{k_z} - 1$ denote the marked time point contained in job $\tilde{J}_z \in \mathbf{I}_{opt}$, for $k_z = 1, 2, \ldots, k-1$, then

$$\mathbb{E}[\sum_{t:t\in[a_z,f_z)} u_t] \geq \frac{1}{2}u_{f_{k_z}-1} \geq \frac{1}{2}v(J_{z+1}) \times \gamma, \tag{6.74}$$

where $v(J_{k_z+1}) \geq v(\tilde{J}_z)$. For (6.72), (6.73) and (6.74) to satisfy constraint (6.45) requires

$$\frac{1}{2}\gamma \geq 1 \qquad \Leftrightarrow \qquad \gamma \geq 2.$$

Therefore, the competitive ratio for the Cooperative Greedy algorithm is 2.

CHAPTER 7

CONCLUSION

This dissertation considers two problems in the broad category of dynamic online resource allocation problems: multi-objective sequential stochastic assignment problems and online interval scheduling problems. For multi-objective sequential stochastic assignment problems, we provide a complete asymptotic analysis for Pareto optimal policies. We consider three classes of Pareto optimal policies and prove that they all achieve the same asymptotic objective values. Convergence rates of these three classes of Pareto optimal policies are also provided for comparison. For online interval scheduling problems, we consider both adversarial and stochastic job sequences. For adversarial online interval scheduling problems, we provide two classes of Greedy algorithms and compute their competitive ratios for scheduling C-benevolent jobs on multiple weighted and unweighted machines. For stochastic online interval scheduling problems, we provide approximation algorithms based on the SSAP optimal policy and the stochastic matching algorithm for both IID and random arrival order job sequences. We also propose a primal-dual framework for analyzing algorithms for both adversarial and stochastic interval scheduling problems. We use strong and weak duality to propose an optimal algorithm for stochastic online interval scheduling problems and compute the competitive ratio of approximation algorithms for adversarial online interval scheduling problems, respectively.

There are several open problems for future research. The online interval scheduling problem in this dissertation considers three classes of job sequences: (a) equal-length jobs, (b) C-benevolent jobs, and (c) memoryless-length jobs. How to extend the approximation algorithms proposed in this dissertation to other classes of job sequences remains an open problem.

The online interval scheduling problem in this dissertation assumes that all machines have the same speed for executing jobs, and hence, job lengths are indifferent with respect to the machine assigned. However, this is not

always the case. One related research topic is interval scheduling on related machines. Generalizing results in this dissertation to the case of taking the speed of machines into consideration is another open problem.

Another interesting research topic is multi-objective online interval scheduling problems. For multi-objective online interval scheduling problems with stochastic job sequences, simply combining results in Chapters 2 and 5 provides a straightforward and naive solution. Other potential approaches include game theory approaches, where each objective can be considered as a player and the online interval scheduling problem can be formulated as a stochastic game. The feasibility of game theory approaches and the relationship between Nash equilibria and Pareto optimal policies remain open problems.

APPENDIX A

PROOFS FOR CHAPTER 2

A.1 Useful Lemmas

Wald's equation [71]: Let $X_1, X_2, \ldots$ be IID random variables and N be a stopping time with respect to $X_i,\ i = 1, 2 \ldots$. If $\mathbb{E}[X_i] < +\infty$ and $\mathbb{E}[N] < +\infty$, then

$$\mathbb{E}[\sum_{i=1}^{N} X_i] = \mathbb{E}[X_i]\mathbb{E}[N].$$

Lemma 1 [25]: For a sequence of IID Bernoulli trials $(X_t)_{1\leq t\leq n}$ with the success probability of p_1, let U_r be the stopping time with respect to $(X_t)_{1\leq t\leq n}$ for the event of obtaining r successes. Let $N_r = \min(U_r, n)$, then for $0 < p_2 < p_1 < 1$,

$$\lim_{n\to+\infty} \frac{\mathbb{E}[N_{\lfloor np_2 \rfloor}]}{n} = \frac{p_2}{p_1}.$$

A.2 Proof of Theorem 2

First, we provide a lower bound for the optimal asymptotic expected weighted reward per task using a feasible policy. Then, we provide an upper bound for this optimal asymptotic expected weighted reward per task using order statistics, which is shown to be the same as the lower bound.

Let S_T^* denote the optimal expected weighted reward per task with T tasks to be assigned and a selectee capacity of $\eta = \lfloor T(1-\zeta) \rfloor$, then $\rho_w^\zeta(\mathbf{w}) = \lim_{T\to+\infty} S_T^*$. By Corollary 1, $S_T^* = \frac{1}{T}\sum_{i=T-\eta+1}^{T} a_{i,0}$. Define random variables

Y_t and Z_t for $\epsilon \geq 0$,

$$Y_t = \begin{cases} \mathcal{G}(t), & \text{if } \mathcal{G}(t) > G_{l+1}, \\ \mathcal{G}(t), & \text{with probability } (q+\epsilon) \text{ if } \mathcal{G}(t) = G_{l+1}, \\ 0, & \text{otherwise,} \end{cases} \qquad t = 1, 2, \ldots, T, \tag{A.1}$$

$$Z_t = \begin{cases} 1, & \text{if } Y_t > 0, \\ 0, & \text{otherwise,} \end{cases} \qquad t = 1, 2, \ldots, T. \tag{A.2}$$

Define $U_\eta \triangleq \min\{k \in \mathbb{Z} : \sum_{t=1}^{k} Z_t = \eta\}$ ($\min \varnothing = +\infty$) and $N_\eta \triangleq \min(U_\eta, T)$. Therefore, N_η is a stopping time with respect to $\{Z_t\}_{t=1}^{T}$ (or $\{Y_t\}_{t=1}^{T}$).

Consider a feasible policy Φ_{LB} (not necessarily optimal) that assigns a task with a combined value $\mathcal{G}(t)$ as follows: the task is assigned to a worker if $\mathcal{G}(t) > G_{l+1}$ and rejected if $\mathcal{G}(t) \leq G_l$. If $\mathcal{G}(t) = G_{l+1}$, the task is assigned to a worker with probability $q + \epsilon_1$ for $\epsilon_1 > 0$ small (ϵ_1 is strictly greater than zero as opposed to $\epsilon \geq 0$ defined in (A.1) and (A.2)) and q given by (2.12). This policy continues to assign tasks in this manner until the number of remaining workers is equal to the number of remaining tasks. At this time, every arriving task is assigned to a worker who has a success rate of one. Let S_1 denote the weighted reward per task under policy Φ_{LB}. Since Φ_{LB} may not be optimal, $\mathbb{E}[S_1] \leq S_T^*$.

Consider another policy Φ'_{LB} (not necessarily optimal) making assignments such that $X_t^{\Phi'_{LB}} = Z_t$, for $t = 1, 2, \ldots, N_\eta$ with $\epsilon = \epsilon_1 > 0$ in (A.1) and (A.2), and if $N_\eta < T$, $X_t^{\Phi'_{LB}} = 0$, for $t = N_\eta + 1, \ldots, T$. Let S_2 denote the weighted reward per task under policy Φ'_{LB}, then $S_2 = \frac{1}{T}\sum_{t=1}^{N_\eta} Y_t$. The probability for a task (among the first N_η tasks) being assigned to a worker under policy Φ'_{LB} is $P_2 = \sum_{k=l+2}^{L} p_{\mathcal{G}}(G_k) + (q+\epsilon_1)p_{\mathcal{G}}(G_{l+1})$, and the tasks assigned to a worker all have a realized combined value greater than or equal to G_{l+1}. Note that $\{Y_t\}_{t=1}^{T}$ are IID. Since N_η is a stopping time with respect to $\{Y_t\}_{t=1}^{T}$,

using Wald's equation and *Lemma 1* (see Appendix A.1),

$$\begin{aligned}\lim_{T\to+\infty}\mathbb{E}[S_2] &= \lim_{T\to+\infty}\frac{\mathbb{E}[N_\eta]}{T}\mathbb{E}[Y_t] \\ &= \frac{1-\zeta}{P_2}\mathbb{E}[Y_t] \\ &= \frac{1-\zeta}{P_2}\left(\left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k)\right) + (q+\epsilon_1)G_{l+1}p_{\mathcal{G}}(G_{l+1})\right). \quad \text{(A.3)}\end{aligned}$$

Next, we show the difference between S_1 and S_2 (denoted by $D = S_1 - S_2$) converges to zero in expectation as $T \to +\infty$. By the definition of D,

$$D = \begin{cases} 0, & \text{if } U_\eta \le T, \\ \frac{1}{T}\sum_{k=1}^{\eta-\sum_{t=1}^{T} Z_t} \mathcal{G}(t_k), & \text{if } U_\eta > T, \end{cases} \quad \text{(A.4)}$$

where $\{t_k\}$ index those tasks assigned to a worker who has a success rate of one under policy Φ_{LB} but not under policy Φ'_{LB}, and hence $\mathcal{G}(t_k) \le G_{l+1}$ for all $1 \le k \le \eta - \sum_{t=1}^{T} Z_t$. By the definition of q, $P_2 = 1-\zeta+\epsilon_1 p_{\mathcal{G}}(G_{l+1}) > 1-\zeta$. Since $\{Z_t\}_{t=1}^{T}$ are IID with $E[Z_t] = P_2$, by the weak law of large numbers,

$$\begin{aligned}\lim_{T\to+\infty}\mathbb{P}(U_\eta > T) &= \lim_{T\to+\infty}\mathbb{P}\left(\frac{1}{T}\sum_{t=1}^{T} Z_t < \frac{\eta}{T}\right) \\ &\le \lim_{T\to+\infty}\mathbb{P}\left(\frac{1}{T}\sum_{t=1}^{T} Z_t < 1-\zeta\right) \\ &= \lim_{T\to+\infty}\mathbb{P}\left(\frac{1}{T}\sum_{t=1}^{T} Z_t < P_2 - \epsilon_1 p_{\mathcal{G}}(G_{l+1})\right) = 0.\end{aligned}$$

Therefore, the expectation of D defined by (A.4) is

$$\begin{aligned}\lim_{T\to+\infty}\mathbb{E}[D] &= \lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\sum_{k=1}^{\eta-\sum_{t=1}^{T} Z_t} \mathcal{G}(t_k)]\mathbb{P}(U_\eta > T) \\ &\le \lim_{T\to+\infty}\frac{\eta G_{l+1}}{T}\mathbb{P}(U_\eta > T) \\ &\le (1-\zeta)G_{l+1}\lim_{T\to+\infty}\mathbb{P}(U_\eta > T) = 0,\end{aligned}$$

where the first inequality follows from $\eta - \sum_{t=1}^{T} Z_t \le \eta$ and $\mathcal{G}(t_k) \le G_{l+1}$. Therefore, $\lim_{T\to+\infty}\mathbb{E}[S_1] = \lim_{T\to+\infty}\mathbb{E}[S_2] = (1-\zeta)\mathbb{E}[Y_t]/P_2$. Moreover,

taking the limit of (A.3) as $\epsilon_1 \to 0$,

$$\begin{aligned}\lim_{\epsilon_1 \to 0} \lim_{T \to +\infty} \mathbb{E}[S_1] &= \lim_{\epsilon_1 \to 0} \frac{1-\zeta}{P_2} \mathbb{E}[Y_t] \\ &= \mathbb{E}[Y_t] \\ &= \left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k) \right) + q G_{l+1} p_{\mathcal{G}}(G_{l+1}) \\ &\leq \lim_{T \to +\infty} S_T^* = \rho_w^{\zeta}(\mathbf{w}), \end{aligned} \tag{A.5}$$

which provides a lower bound for the optimal asymptotic expected weighted reward per task.

Second, we establish an upper bound for the optimal asymptotic expected weighted reward per task. Since the order statistics for the sequence of combined values $\{\mathcal{G}(t)\}_{t=1}^{T}$ are denoted by $\hat{\mathcal{G}}_T^{(1)} \leq \ldots \leq \hat{\mathcal{G}}_T^{(T)}$, then $S_T^* \leq \frac{1}{T}\mathbb{E}[\sum_{t=T-\eta+1}^{T} \hat{\mathcal{G}}_T^{(t)}]$. Let $\epsilon = 0$ in (A.1) and (A.2), and define $N_T \triangleq \sum_{t=1}^{T} Z_t$. Then, N_T is the number of ones in the sequence of $\{Z_t\}_{t=1}^{T}$ and $\mathbb{E}[N_T] = T(1 - F_{\mathcal{G}}(G_{l+1}) + q p_{\mathcal{G}}(G_{l+1})) = T(1-\zeta)$, $\lfloor \mathbb{E}[N_T] \rfloor = \eta$. We show that $\frac{1}{T}\mathbb{E}[\sum_{t=T-\eta+1}^{T} \hat{\mathcal{G}}_T^{(t)} - \sum_{t=1}^{T} Y_t] \leq 0$. Then,

$$\begin{aligned} &\frac{1}{T}\mathbb{E}[\sum_{t=T-\eta+1}^{T} \hat{\mathcal{G}}_T^{(t)} - \sum_{t=1}^{T} Y_t] \\ \overset{(a)}{=} &\frac{1}{T}\left(\sum_{N_T \leq \eta} (\sum_{t=T-\eta+1}^{T} \hat{\mathcal{G}}_T^{(t)} - \sum_{t=1}^{T} Y_t) P_\gamma + \sum_{N_T > \eta} (\sum_{t=T-\eta+1}^{T} \hat{\mathcal{G}}_T^{(t)} - \sum_{t=1}^{T} Y_t) P_\gamma \right) \\ \overset{(b)}{=} &\frac{1}{T}\left(\sum_{N_T \leq \eta} (\hat{\mathcal{G}}_T^{(T-\eta+1)} + \ldots + \hat{\mathcal{G}}_T^{(T-N_T)}) P_\gamma - \sum_{N_T > \eta} (\hat{\mathcal{G}}_T^{(T-N_T+1)} + \ldots + \hat{\mathcal{G}}_T^{(T-\eta)}) P_\gamma \right) \end{aligned} \tag{A.6}$$

$$\begin{aligned} \overset{(c)}{\leq} &\frac{G_{l+1}}{T}\left(\sum_{N_T \leq \eta} (\eta - N_T) P_\gamma - \sum_{N_T > \eta} -(\eta - N_T) P_\gamma \right) \\ = &\frac{G_{l+1}}{T}\mathbb{E}[\eta - N_T] \leq 0. \end{aligned}$$

Here, equality (a) follows from expanding the expectation by conditioning on whether $N_T \leq \eta$, with P_γ denoting the pmf for the sequence $\{\mathcal{G}(t)\}_{t=1}^{T}$; equality (b) follows from the definition of $\{Y_t\}_{t=1}^{T}$ (A.1), and hence, when $N_T \leq \eta$, there are $\eta - N_T$ more terms in $\sum_{t=T-\eta+1}^{T} \hat{\mathcal{G}}_T^{(t)}$ than in $\sum_{t=1}^{T} Y_t$ and

when $N_T > \eta$, there are $N_T - \eta$ more terms in $\sum_{t=1}^{T} Y_t$ than in $\sum_{t=T-\eta+1}^{T} \hat{\mathcal{G}}_T^{(t)}$; inequality (c) follows from (b), and in (A.6) each remaining term in the first sum is less than or equal to G_{l+1} while each remaining term in the second sum is greater than or equal to G_{l+1}. Therefore,

$$S_T^* \leq \frac{1}{T}\mathbb{E}[\sum_{t=T-\eta+1}^{T} \hat{\mathcal{G}}_T^{(t)}] \leq \frac{1}{T}\mathbb{E}[\sum_{t=1}^{T} Y_t] = \mathbb{E}[Y_t] = \left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k)\right) + qG_{l+1}p_{\mathcal{G}}(G_{l+1}). \tag{A.7}$$

Since the lower bound in (A.5) and the upper bound in (A.7) are the same, we have

$$\rho_w^{\zeta}(\mathbf{w}) = \lim_{T\to+\infty} S_T^* = \left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k)\right) + qG_{l+1}p_{\mathcal{G}}(G_{l+1}), \tag{A.8}$$

which completes the proof. ■

A.3 Proof of Corollary 2

From Corollary 1 and Theorem 2, the asymptotic expected weighted reward per task under policy ($\Phi 1$) is optimal and given by

$$\lim_{T\to+\infty} R_w(\Phi 1) = \lim_{T\to+\infty} \frac{1}{T} \sum_{i=\lceil T\zeta\rceil+1}^{T} a_{i,0} = \left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k)\right) + qG_{l+1}p_{\mathcal{G}}(G_{l+1}). \tag{A.9}$$

Choose $F_{\mathcal{G}}(G_l) < \theta' < \theta < F_{\mathcal{G}}(G_{l+1})$. We substitute ζ with θ and θ', respectively. From (A.9),

$$\lim_{T\to+\infty} \frac{1}{T} \sum_{i=\lceil T\theta\rceil+1}^{T} a_{i,0} = \left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k)\right) + q_{\theta}G_{l+1}p_{\mathcal{G}}(G_{l+1}), \tag{A.10}$$

and

$$\lim_{T\to+\infty} \frac{1}{T} \sum_{i=\lceil T\theta'\rceil+1}^{T} a_{i,0} = \left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k)\right) + q'_{\theta}G_{l+1}p_{\mathcal{G}}(G_{l+1}), \tag{A.11}$$

where $q_\theta = \frac{F_{\mathcal{G}}(G_{l+1})-\theta}{p_{\mathcal{G}}(G_{l+1})}$ and $q'_\theta = \frac{F_{\mathcal{G}}(G_{l+1})-\theta'}{p_{\mathcal{G}}(G_{l+1})}$. Taking the difference between (A.10) and (A.11) leads to

$$\lim_{T\to+\infty} \frac{1}{T} \sum_{i=\lceil T\theta' \rceil+1}^{\lceil T\theta \rceil} a_{i,0} = (q'_\theta - q_\theta) G_{l+1} p_{\mathcal{G}}(G_{l+1}) = (\theta - \theta') G_{l+1}. \tag{A.12}$$

For T sufficiently large, $\lceil T\theta \rceil > \lceil T\theta' \rceil + 1$. Since the threshold values $a_{i,0}$ are monotonically increasing with respect to i (from (2.5)), then

$$\frac{1}{T} \sum_{i=\lceil T\theta' \rceil+1}^{\lceil T\theta \rceil} a_{i,0} \leq \frac{\lceil T\theta \rceil - \lceil T\theta' \rceil}{T} a_{\lceil T\theta \rceil,0}, \tag{A.13}$$

which follows by substituting the largest threshold $a_{\lceil T\theta \rceil,0}$ for each $\{a_{i,0}\}_{i=\lceil T\theta' \rceil+1}^{\lceil T\theta \rceil}$ on the left-hand side of (A.13). Dividing both sides by $(\theta - \theta') > 0$, taking the limit of (A.13) as $T \to +\infty$, and using the result in (A.12) lead to

$$G_{l+1} \leq \lim_{T\to+\infty} \inf a_{\lceil T\theta \rceil,0}. \tag{A.14}$$

For the reverse direction, following the same argument, it can be shown the upper bound of the limit of the threshold value is

$$\lim_{T\to+\infty} \sup a_{\lceil T\theta \rceil,0} \leq G_{l+1}. \tag{A.15}$$

Combining (A.14) and (A.15), the result is immediate. ■

A.4 Proof of Lemma 1

Before we prove Lemma 1, we first compute the limit of $b^j_{\lceil T\theta \rceil,0}$ as $T \to +\infty$ for a fixed $F_{\mathcal{G}}(G_l) < \theta < F_{\mathcal{G}}(G_{l+1})$, $l = 0, 1, \ldots, L-1$, and $j = 1, 2 \ldots, n$. These limit values will be used to prove the uniform convergence of $b^j_{\lceil T\theta \rceil,0}$ in the compact interval $I^l_{\epsilon_1,\epsilon_2}$. By Corollary 2, the sequence of random variables $\hat{\mathcal{G}}_T^{(\lceil T\theta \rceil)}$ (indexed by T) converges as

$$\lim_{T\to+\infty} \mathbb{E}[\hat{\mathcal{G}}_T^{(\lceil T\theta \rceil)}] = \lim_{T\to+\infty} a_{\lceil T\theta \rceil,0} = G_{l+1}, \tag{A.16}$$

for $F_{\mathcal{G}}(G_l) < \theta < F_{\mathcal{G}}(G_{l+1})$ and $l = 0, 1, \ldots, L-1$. Lemma 11 provides the convergence type of $\hat{\mathcal{G}}_T^{(\lceil T\theta \rceil)}$ for a fixed θ.

Lemma 11. *The sequence of random variables $\hat{\mathcal{G}}_T^{(\lceil T\theta \rceil)}$ converges to G_{l+1} as $T \to +\infty$ both in mean square and in probability, for $F_{\mathcal{G}}(G_l) < \theta < F_{\mathcal{G}}(G_{l+1})$ and $l = 0, 1, \ldots, L-1$.*

Proof: First we prove

$$\lim_{T \to +\infty} \mathbb{E}[(\hat{\mathcal{G}}_T^{(\lceil T\theta \rceil)})^2] = G_{l+1}^2. \tag{A.17}$$

Then combining (A.16) and (A.17), convergence in mean square can be established, which implies convergence in probability.

To prove (A.17), consider an auxiliary SSAP instance for WOSA-**w** discussed in Theorem 2, with T tasks to be assigned to homogeneous workers. There are $\lfloor T(1-\zeta) \rfloor$ workers (without loss of generality, set their success rates as one); we assume $T - \lfloor T(1-\zeta) \rfloor$ virtual workers with a success rate of zero. For this auxiliary SSAP instance, each task has a combined value $\bar{\mathcal{G}}(t) = \mathcal{G}(t)^2$ (random variable), with realized value $\bar{\gamma}_t = \gamma_t^2$. Note that $\bar{\mathcal{G}}(t)$ preserves the order of $\mathcal{G}(t)$ (i.e., $\bar{\mathcal{G}}(t_1) > \bar{\mathcal{G}}(t_2)$ if and only if $\mathcal{G}(t_1) > \mathcal{G}(t_2)$). Then, $\bar{\mathcal{G}}(t)$ is discrete and takes values $0 < G_1^2 < G_2^2 < \ldots < G_L^2$, with cdf and pmf given by $F_{\bar{\mathcal{G}}}(\bar{\gamma}) = F_{\mathcal{G}}(\sqrt{\bar{\gamma}})$ and $p_{\bar{\mathcal{G}}}(\bar{\gamma}) = p_{\mathcal{G}}(\sqrt{\bar{\gamma}})$, respectively. Let $\bar{\Phi}$ denote the SSAP optimal policy for this instance. Then from Theorem 1, policy $\bar{\Phi}$ is determined by the threshold values $a'_{i,0}, i = 1, 2, \ldots, T$ defined by (2.6), which are the expected value of the i^{th} smallest value of $\bar{\mathcal{G}}(t)$, i.e.,

$$a'_{i,0} = \mathbb{E}[(\hat{\mathcal{G}}_T^{(i)})^2]. \tag{A.18}$$

Now we consider the optimal asymptotic expected reward per task for this auxiliary SSAP instance as $T \to +\infty$. From Theorem 2,

$$\lim_{T \to +\infty} \frac{1}{T} \sum_{i=\lceil T\zeta \rceil + 1}^{T} a'_{i,0} = \left(\sum_{k=l+2}^{L} G_k^2 p_{\mathcal{G}}(G_k) \right) + q G_{l+1}^2 p_{\mathcal{G}}(G_{l+1}),$$

with q defined by (2.12). Applying Corollary 2 and using (A.18),

$$\lim_{T \to +\infty} \mathbb{E}[(\hat{\mathcal{G}}_T^{(\lceil T\theta \rceil)})^2] = \lim_{T \to +\infty} a'_{\lceil T\theta \rceil, 0} = G_{l+1}^2,$$

which proves (A.17). Using (A.16) and (A.17), the limit of the mean square difference is given by

$$\begin{aligned}\lim_{T\to+\infty}\mathbb{E}[(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}-G_{l+1})^2] &= \lim_{T\to+\infty}\mathbb{E}[(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)})^2-2G_{l+1}\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}+G_{l+1}^2]\\ &= G_{l+1}^2-2G_{l+1}^2+G_{l+1}^2=0. \end{aligned} \tag{A.19}$$

Therefore, $\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}$ converges to G_{l+1} in mean square, which implies convergence in probability.

Lemma 12. *The limit of $b^j_{\lceil T\theta\rceil,0}$ defined by (2.8) is*

$$\lim_{T\to+\infty} b^j_{\lceil T\theta\rceil,0}=\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}],$$

for $F_{\mathcal{G}}(G_l)<\theta<F_{\mathcal{G}}(G_{l+1})$, $l=0,1,\ldots,L-1$, and $j=1,2,\ldots,n$.

Proof: From Lemma 11, $\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}$ converges in probability to G_{l+1}. Then for any $\epsilon>0$,

$$\begin{aligned}&\lim_{T\to+\infty}\mathbb{P}(|\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}-G_{l+1}|\geq\epsilon)=0\\ \Rightarrow&\lim_{T\to+\infty}\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}\geq G_{l+1}+\epsilon,\text{ or }\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}\leq G_{l+1}-\epsilon)=0\\ \Rightarrow&\lim_{T\to+\infty}\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}\neq G_{l+1})=0\text{ and }\lim_{T\to+\infty}\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}=G_{l+1})=1,\end{aligned} \tag{A.20}$$

where the last line follows from $\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}$ being a discrete random variable and ϵ arbitrarily small. Since $\mathcal{A}_j(t)$ is bounded above by $A_M<+\infty$, then from the definition of $b^j_{\lceil T\theta\rceil,0}$ (2.8),

$$\begin{aligned}\lim_{T\to+\infty} b^j_{\lceil T\theta\rceil,0}&=\lim_{T\to+\infty}\mathbb{E}\left[\mathbb{E}[\hat{\mathcal{A}}_T^{(j)(\lceil T\theta\rceil)}|\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}]\right]\\ &=\lim_{T\to+\infty}\mathbb{E}[\hat{\mathcal{A}}_T^{(j)(\lceil T\theta\rceil)}|\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}=G_{l+1}]\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}=G_{l+1})\\ &\quad+\lim_{T\to+\infty}\mathbb{E}[\hat{\mathcal{A}}_T^{(j)(\lceil T\theta\rceil)}|\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}\neq G_{l+1}]\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}\neq G_{l+1})\\ &\overset{(A.20)}{=}\mathbb{E}[\hat{\mathcal{A}}_T^{(j)(\lceil T\theta\rceil)}|\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}=G_{l+1}]\\ &=\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}],\end{aligned} \tag{A.21}$$

where the last equality follows from that $\mathcal{A}_j(t)$ and $\mathcal{G}(t)$ are both IID.

Proof: First we show $a_{\lceil T\theta\rceil,0}$ converges uniformly to G_{l+1} as $T \to +\infty$ for $\theta \in I^l_{\epsilon_1,\epsilon_2}$. From (2.6), $a_{i,0}$ are uniformly bounded above by G_L for $i = 1, 2, \ldots, T$. Define the following sequence of functions for $0 < \theta \leq 1$ indexed by T,

$$g_T(\theta) \triangleq a_{\lceil T\theta\rceil,0} = a_{i,0}, \text{ for } \frac{i-1}{T} < \theta \leq \frac{i}{T},\ i = 1, 2, \ldots, T. \tag{A.22}$$

From this definition, $g_T(\theta)$ is a left-continuous step function and $g_T(\theta)$ is non-decreasing since $\{a_{i,0}\}$ are monotonically increasing in i. Now consider $g_T(\theta)$ over the compact interval $I^l_{\epsilon_1,\epsilon_2}$, for any $\epsilon_1 > 0$, $\epsilon_2 > 0$. From Corollary 2, $g_T(\theta)$ converge pointwise to G_{l+1} as $T \to +\infty$ over $I^l_{\epsilon_1,\epsilon_2}$. Moreover,

$$\begin{aligned}
&\lim_{T\to+\infty} \sup_{\theta\in I^l_{\epsilon_1,\epsilon_2}} |g_T(\theta) - G_{l+1}| \\
=&\lim_{T\to+\infty} \left(\max\left(|g_T(F_{\mathcal{G}}(G_l) + \epsilon_1) - G_{l+1}|, |g_T(F_{\mathcal{G}}(G_{l+1}) - \epsilon_2) - G_{l+1}|\right)\right) \\
=&\max(\lim_{T\to+\infty} |g_T(F_{\mathcal{G}}(G_l) + \epsilon_1) - G_{l+1}|, \lim_{T\to+\infty} |g_T(F_{\mathcal{G}}(G_{l+1}) - \epsilon_2) - G_{l+1}|) \\
=&0,
\end{aligned}$$

where the first equality follows from $g_T(\theta)$ being non-decreasing over the compact interval $I^l_{\epsilon_1,\epsilon_2}$. Therefore, $g_T(\theta)$ converges uniformly to G_{l+1} in $I^l_{\epsilon_1,\epsilon_2}$. Then, $a_{\lceil T\theta\rceil,0}$ converges uniformly to G_{l+1} as $T \to +\infty$ over the interval $I^l_{\epsilon_1,\epsilon_2}$.

Applying the same arguments to the auxiliary SSAP instance defined in the proof of Lemma 11, then $a'_{\lceil T\theta\rceil,0}$ (A.18) converges uniformly to G^2_{l+1} for $\theta \in I^l_{\epsilon_1,\epsilon_2}$. Moreover, from (A.19), $\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}$ converges uniformly to G_{l+1} in mean square as $T \to +\infty$ for $\theta \in I^l_{\epsilon_1,\epsilon_2}$. Moreover, uniform convergence in mean square implies uniform convergence in probability. Therefore, for any $\epsilon > 0$, there exists an $N_\epsilon \in \mathbb{Z}^+$ (depending only on ϵ) such that for $T > N_\epsilon$ and any $\theta \in I^l_{\epsilon_1,\epsilon_2}$,

$$\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}) < \epsilon, \tag{A.23}$$

which follows from (A.20). Since $\mathcal{A}_j(t)$ is bounded above by $A_M < +\infty$, then for $T > N_\epsilon$

$$\begin{aligned}
&\sup_{\theta\in I^l_{\epsilon_1,\epsilon_2}} |b^j_{\lceil T\theta\rceil,0} - \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{l+1}]| \\
\overset{(2.8)}{=}&\sup_{\theta\in I^l_{\epsilon_1,\epsilon_2}} |\mathbb{E}[\hat{\mathcal{A}}^{(j)(\lceil T\theta\rceil)}_T|\hat{\mathcal{G}}^{(\lceil T\theta\rceil)}_T \neq G_{l+1}] - \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{l+1}]|\mathbb{P}(\hat{\mathcal{G}}^{(\lceil T\theta\rceil)}_T \neq G_{l+1}) \\
\leq& 2A_M\mathbb{P}(\hat{\mathcal{G}}^{(\lceil T\theta\rceil)}_T \neq G_{l+1}) < 2A_M\epsilon \to 0,
\end{aligned}$$

and hence $b^j_{\lceil T\theta\rceil,0}$ converges uniformly to $\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{L+1}]$ for $\theta \in I^l_{\epsilon_1,\epsilon_2}$. ■

A.5 Proof of Theorem 3

For $j = 1, 2, \ldots, n$, from (2.10),

$$\rho^\zeta_j(\mathbf{w}) = \lim_{T\to+\infty} r_j(\Phi 1) = \lim_{T\to+\infty} \frac{1}{T} \sum_{i=\lceil T\zeta\rceil+1}^{T} b^j_{i,0}.$$

In the following, we compute $\rho^\zeta_j(\mathbf{w})$ by summing up the limit of $b^j_{i,0}$, for $i = \lceil T\zeta\rceil + 1, \ldots, T$. First, we need to use the uniform convergence of $b^j_{i,0}$ provided in Lemma 1 to interchange the order of summation and limit.

For any $\epsilon_1 > 0,\ \epsilon_2 > 0$, define compact intervals

$$\begin{aligned}
I &\triangleq [\lceil T\zeta\rceil + 1, T] \subset \mathbb{Z}^+, \\
I_{l+1}(\epsilon_1, \epsilon_2) &\triangleq [\lfloor T(\zeta + \epsilon_1)\rfloor + 1, \lceil T(F_{\mathcal{G}}(G_{l+1}) - \epsilon_2)\rceil] \subset \mathbb{Z}^+, \\
I_m(\epsilon_1, \epsilon_2) &\triangleq [\lfloor T(F_{\mathcal{G}}(G_{m-1}) + \epsilon_1)\rfloor + 1, \lceil T(F_{\mathcal{G}}(G_m) - \epsilon_2)\rceil] \subset \mathbb{Z}^+, \quad \text{(A.24)}
\end{aligned}$$

for $m = l + 2, l + 3, \ldots, L$. Then,

$$I_e \triangleq I \setminus \bigcup_{m=l+1,\ldots,L} I_m(\epsilon_1, \epsilon_2), \quad \text{(A.25)}$$

where I_e is the difference between I and $\bigcup_{m=l+1,\ldots,L} I_m(\epsilon_1, \epsilon_2)$. From Lemma 1, $b^j_{\lceil T\theta\rceil,0}$ converges uniformly to $\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_m]$ as $T \to +\infty$, for $\lceil T\theta\rceil \in I_m(\epsilon_1, \epsilon_2),\ m = l + 1, \ldots, L$. Therefore, the summation and limit in (2.10) are interchangeable over $I \setminus I_e$.

The limits of the counting measure ($\mu([a,b]) \triangleq \lfloor b \rfloor - \lceil a \rceil + 1$) of these compact intervals normalized by T are

$$\lim_{T\to+\infty} \frac{1}{T}\mu(I_{l+1}(\epsilon_1,\epsilon_2)) = F_{\mathcal{G}}(G_{l+1}) - \epsilon_2 - \zeta - \epsilon_1 \overset{(2.12)}{=} qp_{\mathcal{G}}(G_{l+1}) - (\epsilon_1+\epsilon_2), \tag{A.26}$$

$$\lim_{T\to+\infty} \frac{1}{T}\mu(I_m(\epsilon_1,\epsilon_2)) = F_{\mathcal{G}}(G_m) - \epsilon_2 - F_{\mathcal{G}}(G_{m-1}) - \epsilon_1 = p_{\mathcal{G}}(G_m) - (\epsilon_1+\epsilon_2), \tag{A.27}$$

for $m = l+2, \ldots, L$, and

$$\lim_{T\to+\infty} \frac{1}{T}\mu(I_e) = \lim_{T\to+\infty} \frac{1}{T}\mu(I \setminus \bigcup_{m=l+1,\ldots,L} I_m(\epsilon_1,\epsilon_2)) = (L-l)(\epsilon_1+\epsilon_2). \tag{A.28}$$

Then (2.10) can be written as

$$\begin{aligned}
\lim_{T\to+\infty} \frac{1}{T} \sum_{i=\lceil T\zeta\rceil+1}^{T} b_{i,0}^j &= \lim_{T\to+\infty} \frac{1}{T} \sum_{i=\lceil T\zeta\rceil+1}^{T} \left(\left(\sum_{m=l+1}^{L} \mathbf{1}_{i\in I_m(\epsilon_1,\epsilon_2)} \right) + \mathbf{1}_{i\in I_e} \right) b_{i,0}^j \\
&= \left(\lim_{T\to+\infty} \frac{1}{T} \sum_{m=l+1}^{L} \sum_{i=\lceil T\zeta\rceil+1}^{T} \mathbf{1}_{i\in I_m(\epsilon_1,\epsilon_2)} b_{i,0}^j \right) \\
&\quad + \lim_{T\to+\infty} \frac{1}{T} \sum_{i=\lceil T\zeta\rceil+1}^{T} \mathbf{1}_{i\in I_e} b_{i,0}^j,
\end{aligned} \tag{A.29}$$

where $\mathbf{1}_{i\in I_m(\epsilon_1,\epsilon_2)}$ is the indicator function and the first equality follows from the mutually exclusive and exhaustive partition $I = \left(\bigcup_{m=l+1,\ldots,L} I_m(\epsilon_1,\epsilon_2)\right) \bigcup I_e$.

For the first term of (A.29),

$$
\begin{aligned}
&\lim_{T\to+\infty}\frac{1}{T}\sum_{m=l+1}^{L}\sum_{i=\lceil T\zeta\rceil+1}^{T}\mathbf{1}_{i\in I_m(\epsilon_1,\epsilon_2)}b_{i,0}^{j}\\
=&\sum_{m=l+1}^{L}\lim_{T\to+\infty}\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}\mathbf{1}_{i\in I_m(\epsilon_1,\epsilon_2)}b_{i,0}^{j}\\
\overset{(a)}{=}&\sum_{m=l+1}^{L}\left(\lim_{T\to+\infty}b_{i,0}^{j}\right)\left(\lim_{T\to+\infty}\frac{1}{T}\mu(I_m(\epsilon_1,\epsilon_2))\right)\\
\overset{(A.26),(A.27)}{=}&\left(\sum_{m=l+2}^{L}\left(p_{\mathcal{G}}(G_m)-(\epsilon_2+\epsilon_1)\right)\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_m]\right)\\
&+\Big(qp_{\mathcal{G}}(G_{l+1})-(\epsilon_1+\epsilon_2)\Big)\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}]\\
=&\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)>G_{l+1}]\mathbb{P}_{\mathcal{G}}(\mathcal{G}(t)>G_{l+1})+q\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}]p_{\mathcal{G}}(G_{l+1})\\
&+(\epsilon_1+\epsilon_2)\sum_{m=l+1}^{L}\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_m],
\end{aligned}
\tag{A.30}
$$

where equality (a) follows from the uniform convergence of $b_{i,0}^{j}$ over $i\in I_m(\epsilon_1,\epsilon_2)$ for each $m=l+1,\ldots,L$ and the product law of limit [72]. For simplicity, let $\rho_j^*=\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)>G_{l+1}]\mathbb{P}_{\mathcal{G}}(\mathcal{G}(t)>G_{l+1})+q\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}]p_{\mathcal{G}}(G_{l+1})$.

For the second term of (A.29),

$$
\lim_{T\to+\infty}\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}\mathbf{1}_{i\in I_e}b_{i,0}^{j}\le\lim_{T\to+\infty}\frac{1}{T}\mu(I_e)A_M\overset{(A.28)}{=}(\epsilon_1+\epsilon_2)(L-l)A_M,
\tag{A.31}
$$

where the inequality follows from $b_{\lceil T\theta\rceil,0}^{j}$ uniformly bounded above by $A_M<+\infty$ for $0\le\theta\le1$ (since $\mathcal{A}_j(t)$ is bounded above by $A_M<+\infty$).

From (A.30), for any $\epsilon>0$, there exists $N_\epsilon^1\in\mathbb{Z}^+$, such that for $T>N_\epsilon^1$,

$$
\left|\frac{1}{T}\sum_{m=l+1}^{L}\sum_{i=\lceil T\zeta\rceil+1}^{T}\mathbf{1}_{i\in I_m(\epsilon_1,\epsilon_2)}b_{i,0}^{j}-\rho_j^*-(\epsilon_1+\epsilon_2)\sum_{m=l+1}^{L}\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_m]\right|<\epsilon
$$

$$\Rightarrow |\frac{1}{T}\sum_{m=l+1}^{L}\sum_{i=\lceil T\zeta\rceil+1}^{T}\mathbf{1}_{i\in I_m(\epsilon_1,\epsilon_2)}b_{i,0}^j-\rho_j^*|<(\epsilon_1+\epsilon_2)\left(\sum_{m=l+1}^{L}\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_m]\right)+\epsilon$$
$$<(\epsilon_1+\epsilon_2)(L-l)A_M+\epsilon. \qquad (A.32)$$

From (A.31), there exists $N_\epsilon^2\in\mathbb{Z}^+$, such that for $T>N_\epsilon^2$,

$$|\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}\mathbf{1}_{i\in I_e}b_{i,0}^j|<(\epsilon_1+\epsilon_2)(L-l)A_M+\epsilon. \qquad (A.33)$$

Therefore, for $T>\max(N_\epsilon^1,N_\epsilon^2)$, from (A.32) and (A.33),

$$|\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}b_{i,0}^j-\rho_j^*|\le|\frac{1}{T}\sum_{m=l+1}^{L}\sum_{i=\lceil T\zeta\rceil+1}^{T}\mathbf{1}_{i\in I_m(\epsilon_1,\epsilon_2)}b_{i,0}^j-\rho_j^*|+|\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}\mathbf{1}_{i\in I_e}b_{i,0}^j|$$
$$<2(\epsilon_1+\epsilon_2)(L-l)A_M+2\epsilon, \qquad (A.34)$$

which holds for any $\epsilon_1>0,\ \epsilon_2>0$. Taking $\epsilon_1=\epsilon_2=\epsilon/2$, then (A.34) becomes $(2(L-l)A_M+2)\,\epsilon$, with $2(L-l)A_M+2$ finite and constant. Therefore, $\rho_j^\zeta(\mathbf{w})=\lim_{T\to+\infty}\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}b_{i,0}^j=\rho_j^*$. ■

A.6 Proof of Theorem 4

Let $f_{\Phi 2}(\zeta)$ denote the asymptotic expected weighted reward per task for WOSA-**w** under policy ($\Phi 2$) (not necessarily optimal). Let $\zeta(n)$ denote the non-selectee ratio after n task assignments, given by

$$\zeta(n)=1-\frac{\lfloor(1-\zeta)T\rfloor-\sum_{t=1}^{n}X_t^{\Phi 2}}{T-n},\ n=1,2,\ldots,T. \qquad (A.35)$$

First, we show that $F_{\mathcal{G}}(G_l)\le\zeta(\lfloor\nu_\pi T\rfloor)\le F_{\mathcal{G}}(G_{l+1})$ hold regardless of the first $\lfloor\nu_\zeta T\rfloor$ assignments. From the definition of ν_ζ (2.15), we have $\nu_\zeta\le\zeta$. Since $0\le\sum_{t=1}^{\lfloor\nu_\zeta T\rfloor}X_t^{\Phi 2}\le\lfloor\nu_\zeta T\rfloor\le\nu_\zeta T$ for any $T\in\mathbb{Z}^+$, then

$$1-\frac{\lfloor(1-\zeta)T\rfloor}{T-\lfloor\nu_\zeta T\rfloor}\le\zeta(\lfloor\nu_\zeta T\rfloor)\le1-\frac{\lfloor(1-\zeta)T\rfloor-\lfloor\nu_\zeta T\rfloor}{T-\lfloor\nu_\zeta T\rfloor}.$$

Moreover, since $x - 1 \leq \lfloor x \rfloor \leq x$ for all $x > 0$, then

$$\nu_\zeta < \frac{F_{\mathcal{G}}(G_{l+1}) - \zeta}{F_{\mathcal{G}}(G_{l+1})} \Rightarrow 1 - \frac{\lfloor (1-\zeta)T \rfloor - \lfloor \nu_\zeta T \rfloor}{T - \lfloor \nu_\zeta T \rfloor} \leq F_{\mathcal{G}}(G_{l+1}), \tag{A.36}$$

and

$$\nu_\zeta \leq \frac{\zeta - F_{\mathcal{G}}(G_l)}{1 - F_{\mathcal{G}}(G_l)} \Rightarrow 1 - \frac{\lfloor (1-\zeta)T \rfloor}{T - \lfloor \nu_\zeta T \rfloor} \geq F_{\mathcal{G}}(G_l). \tag{A.37}$$

Therefore, $F_{\mathcal{G}}(G_l) \leq \zeta(\lfloor \nu_\zeta T \rfloor) \leq F_{\mathcal{G}}(G_{l+1})$ always hold. This condition together with Theorem 2 guarantees the asymptotic expected weighted reward per task for WOSA-**w** under policy ($\Phi 1$) after the first $\lfloor \nu_\zeta T \rfloor$ task assignments is linear with respect to the non-selectee ratio $\zeta(\lfloor \nu_\zeta T \rfloor)$.

Since WOSA-**w** can be formulated as a MDP (see *Section 4.1* by [65]), the expected weighted reward per task can be broken up into two parts: the expected weighted reward per task for assigning the first $\lfloor \nu_\zeta T \rfloor$ tasks, and the expected weighted reward per task for assigning the remaining $T - \lfloor \nu_\zeta T \rfloor$ tasks. These two parts are independent conditional on the non-selectee ratio $\zeta(\lfloor \nu_\zeta T \rfloor)$, which leads to

$$\begin{aligned} f_{\Phi 2}(\zeta) &= \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2} \mathcal{G}(t) + \sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 2} \mathcal{G}(t)] \\ &= \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2} \mathcal{G}(t)] + \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 1} \mathcal{G}(t)] \\ &= \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2} \mathcal{G}(t)] + (1 - \nu_\zeta) \mathbb{E}[\rho_w^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})], \end{aligned} \tag{A.38}$$

where the last equality follows from Theorem 2 and the expectation of the second term in (A.38) is taken with respect to $\zeta(\lfloor \nu_\zeta T \rfloor)$, which depends on the first $\lfloor \nu_\zeta T \rfloor$ task assignments.

Next, we consider the first term in (A.38). Since the first $\lfloor \nu_\zeta T \rfloor$ tasks have all been assigned in the same manner, $\{X_t^{\Phi 2} \mathcal{G}(t)\}$ and $\{X_t^{\Phi 2}\}$ are both IID for $t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor$, with $\mathbb{E}[X_t^{\Phi 2} \mathcal{G}(t)] = \sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k) + q G_{l+1} p_{\mathcal{G}}(G_{l+1})$ and $\mathbb{E}[X_t^{\Phi 2}] = \sum_{k=l+2}^{L} p_{\mathcal{G}}(G_k) + q p_{\mathcal{G}}(G_{l+1})$. Since $\lfloor \nu_\zeta T \rfloor \to +\infty$ as $T \to +\infty$, then from the weak law of large numbers, $(1/\lfloor \nu_\zeta T \rfloor) \sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2} \mathcal{G}(t)$ and $(1/\lfloor \nu_\zeta T \rfloor) \sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}$ converge to their corresponding expectation in

probability, respectively. Moreover, since $X_t^{\Phi 2}\mathcal{G}(t) \leq G_L$ and $X_t^{\Phi 2} \leq 1$ for $t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor$ are both bounded above, convergence in probability implies convergence in the mean square sense and in mean. Therefore,

$$\begin{aligned}\lim_{T\to+\infty} \frac{1}{T}\mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}\mathcal{G}(t)] &= \nu_\zeta \mathbb{E}[X_t^{\Phi 2}\mathcal{G}(t)] \\ &= \nu_\zeta \left(\left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k) \right) + q G_{l+1} p_{\mathcal{G}}(G_{l+1}) \right), \end{aligned} \tag{A.39}$$

$$\begin{aligned}\lim_{T\to+\infty} \frac{1}{T}\mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}] &= \nu_\zeta \mathbb{E}[X_t^{\Phi 2}] \\ &= \nu_\zeta \left(\left(\sum_{k=l+2}^{L} p_{\mathcal{G}}(G_k) \right) + q p_{\mathcal{G}}(G_{l+1}) \right) \\ &= \nu_\zeta (1-\zeta), \end{aligned} \tag{A.40}$$

and

$$\begin{aligned}\lim_{T\to+\infty} \mathbb{E}[\zeta(\lfloor \nu_\zeta T \rfloor)] &= \lim_{T\to+\infty} 1 - \frac{\lfloor (1-\zeta)T \rfloor - \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}]}{T - \lfloor \nu_\zeta T \rfloor} \\ &= 1 - \frac{(1-\zeta) - \epsilon(1-\zeta)}{(1-\nu_\zeta)} = \zeta. \end{aligned} \tag{A.41}$$

For the second term in (A.38), since (A.36) and (A.37) hold, $\rho_w^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})$ is an affine function of $\zeta(\lfloor \nu_\zeta T \rfloor)$. Then from Theorem 2,

$$\begin{aligned}&\lim_{T\to+\infty} (1-\nu_\zeta)\mathbb{E}[\rho_w^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})] \\ &= \lim_{T\to+\infty} (1-\nu_\zeta)\mathbb{E}[\left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k) \right) + \Big(F_{\mathcal{G}}(G_{l+1}) - \zeta(\lfloor \nu_\zeta T \rfloor) \Big) G_{l+1}] \\ &= \lim_{T\to+\infty} (1-\nu_\zeta) \left(\left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k) \right) + \Big(F_{\mathcal{G}}(G_{l+1}) - \mathbb{E}[\zeta(\lfloor \nu_\zeta T \rfloor)] \Big) G_{l+1} \right) \\ &= (1-\nu_\zeta) \left(\left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k) \right) + (F_{\mathcal{G}}(G_{l+1}) - \zeta) G_{l+1} \right). \end{aligned} \tag{A.42}$$

Substituting (A.39) and (A.42) into (A.38) leads to the asymptotic expected weighted reward per task under policy ($\Phi 2$),

$$f_{\Phi 2}(\zeta) = \left(\sum_{k=l+2}^{L} G_k p_{\mathcal{G}}(G_k) \right) + (F_{\mathcal{G}}(G_{l+1}) - \zeta) G_{l+1},$$

which matches the optimal asymptotic expected weighted reward per task (given in Theorem 2). ■

A.7 Proof of Theorem 5

Let $h_j(\zeta)$ denote the asymptotic expected reward per task for $r_j(\Phi 2)$ (not necessarily optimal) under the SSAP mixed policy ($\Phi 2$), for $j = 1, 2, \ldots, n$. Since WOSA-**w** can be formulated as an MDP (see *Section 4.1* by [65]), $h_j(\zeta)$ can be broken up into two parts: the expected reward per task for assigning the first $\lfloor \nu_\zeta T \rfloor$ tasks and the expected reward per task for assigning the remaining $T - \lfloor \nu_\zeta T \rfloor$ tasks. These two parts are independent conditional on the non-selectee ratio $\zeta(\lfloor \nu_\zeta T \rfloor)$, which is defined by (A.35) as the non-selectee ratio after the first $\lfloor \nu_\zeta T \rfloor$ task assignments. Therefore,

$$\begin{aligned} h_j(\zeta) &= \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2} \mathcal{A}_j(t) + \sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 2} \mathcal{A}_j(t)] \\ &= \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2} \mathcal{A}_j(t)] + \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 1} \mathcal{A}_j(t)] \\ &= \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2} \mathcal{A}_j(t)] + \lim_{T \to +\infty} (1 - \nu_\zeta) \mathbb{E}[\rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})], \quad \text{(A.43)} \end{aligned}$$

where the last equality follows from Theorem 3 and the expectation of the second term is taken with respect to $\zeta(\lfloor \nu_\zeta T \rfloor)$, which depends on the first $\lfloor \nu_\zeta T \rfloor$ assignments. By (A.36) and (A.37), $F_{\mathcal{G}}(G_l) \leq \zeta(\lfloor \nu_\zeta T \rfloor) \leq F_{\mathcal{G}}(G_{l+1})$, which together with Theorem 3 guarantees that $\rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})$ is an affine function of $\zeta(\lfloor \nu_\zeta T \rfloor)$.

Next, we consider the first term in (A.43). Consider $L - l$ random variables defined by the indicator function $\Gamma_t^{(k)} \triangleq \mathbf{1}_{\mathcal{G}(t)=G_k, X_t^{\Phi}=1}$, for $k = l+1, \ldots, L$ and $t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor$. From these definitions, $X_t^{\Phi 2} = \sum_{k=l+1}^{L} \Gamma_t^{(k)}$ for

$t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor$. Since the first $\lfloor \nu_\zeta T \rfloor$ tasks have all been assigned in the same manner, then for each fixed k, $\{\Gamma_t^{(k)}\}$ and $\{\Gamma_t^{(k)} \mathcal{A}_j(t)\}$ are both IID for $t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor$, with expectations given by

$$\begin{aligned}
&\mathbb{E}[\Gamma_t^{(l+1)}] = q p_{\mathcal{G}}(G_{l+1}), \quad \mathbb{E}[\Gamma_t^{(l+1)} \mathcal{A}_j(t)] = q \mathbb{E}[\mathcal{A}_j(t) | \mathcal{G}(t) = G_{l+1}] p_{\mathcal{G}}(G_{l+1}), \\
&\mathbb{E}[\Gamma_t^{(k)}] = p_{\mathcal{G}}(G_k), \qquad\quad \mathbb{E}[\Gamma_t^{(k)} \mathcal{A}_j(t)] = \mathbb{E}[\mathcal{A}_j(t) | \mathcal{G}(t) = G_k] p_{\mathcal{G}}(G_k), \quad \text{(A.44)}
\end{aligned}$$

for $k = l + 2, \ldots, L$. Since $\lfloor \nu_\zeta T \rfloor \to +\infty$ as $T \to +\infty$, then from the weak law of large numbers, $(1/\lfloor \nu_\zeta T \rfloor) \sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} \Gamma_t^{(k)}$ and $(1/\lfloor \nu_\zeta T \rfloor) \sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} \Gamma_t^{(k)} \mathcal{A}_j(t)$ converge to the corresponding expectation in probability, respectively. Moreover, since $\Gamma_t^{(k)} \leq 1$ and $\Gamma_t^{(k)} \mathcal{A}_j(t) \leq A_M$ are both bounded above for $t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor$, convergence in probability implies convergence in the mean square sense and in mean. Therefore,

$$\begin{aligned}
\lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2} \mathcal{A}_j(t)] &= \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} \sum_{k=l+1}^{L} \Gamma_t^{(k)} \mathcal{A}_j(t)] \\
&= \sum_{k=l+1}^{L} \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} \Gamma_t^{(k)} \mathcal{A}_j(t)] \\
&= \nu_\zeta \sum_{k=l+1}^{L} \mathbb{E}[\Gamma_t^{(k)} \mathcal{A}_j(t)] \\
&= \nu_\zeta \Big(\Big(\sum_{k=l+2}^{L} \mathbb{E}[\mathcal{A}_j(t) | \mathcal{G}(t) = G_k] p_{\mathcal{G}}(G_k) \Big) \\
&\quad + q \mathbb{E}[\mathcal{A}_j(t) | \mathcal{G}(t) = G_{l+1}] p_{\mathcal{G}}(G_{l+1}) \Big), \qquad \text{(A.45)} \\
\lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}] &= \lim_{T \to +\infty} \frac{1}{T} \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} \sum_{k=l+1}^{L} \Gamma_t^{(k)}] \\
&= \nu_\zeta \sum_{k=l+1}^{L} \mathbb{E}[\Gamma_t^{(k)}] \\
&= \nu_\zeta \Big(\Big(\sum_{k=l+2}^{L} p_{\mathcal{G}}(G_k) \Big) + q p_{\mathcal{G}}(G_{l+1}) \Big) = \nu_\zeta (1 - \zeta). \\
&\qquad \text{(A.46)}
\end{aligned}$$

For the second term in (A.43), since (A.36) holds, $\rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})$ is an affine function of $\zeta(\lfloor \nu_\zeta T \rfloor)$. Then from Theorem 3,

$$\begin{aligned}
&\lim_{T\to+\infty}(1-\nu_\zeta)\mathbb{E}[\rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})] \\
=&\lim_{T\to+\infty}(1-\nu_\zeta)\mathbb{E}\bigg[\bigg(\sum_{k=l+2}^{L}\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_k]p_{\mathcal{G}}(G_k)\bigg) \\
&+\Big(F_{\mathcal{G}}(G_{l+1})-\zeta(\lfloor \nu_\zeta T \rfloor)\Big)\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}]\bigg] \\
=&\lim_{T\to+\infty}(1-\nu_\zeta)\bigg(\Big(\sum_{k=l+2}^{L}\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_k]p_{\mathcal{G}}(G_k)\Big) \\
&+\Big(F_{\mathcal{G}}(G_{l+1})-\mathbb{E}[\zeta(\lfloor \nu_\zeta T \rfloor)]\Big)\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}]\bigg) \\
=&(1-\nu_\zeta)\bigg(\Big(\sum_{k=l+2}^{L}\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_k]p_{\mathcal{G}}(G_k)\Big) \\
&+(F_{\mathcal{G}}(G_{l+1})-\zeta)\mathbb{E}[\mathcal{A}(t)|\mathcal{G}(t)=G_{l+1}]\bigg). \qquad \text{(A.47)}
\end{aligned}$$

Substituting (A.45) and (A.47) into (A.43) leads to the asymptotic expected reward per task for $r_j(\Phi 2)$,

$$\begin{aligned}
h_j(\zeta) &= \bigg(\sum_{k=l+2}^{L}\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_k]p_{\mathcal{G}}(G_k)\bigg)+q\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}]p_{\mathcal{G}}(G_{l+1}) \\
&= \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)>G_{l+1}]\mathbb{P}_{\mathcal{G}}(\mathcal{G}(t)>G_{l+1})+q\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}]p_{\mathcal{G}}(G_{l+1}),
\end{aligned}$$

which is the same as $\rho_j^{\zeta}(\mathbf{w})$ given by (2.13). ■

A.8 Proof of Theorem 6

Let $\eta = \lfloor T(1-\zeta) \rfloor$. Consider the following auxiliary policy $\Phi 3'$:

$$X_t^{\Phi 3'} = \begin{cases} X_t^{\Phi 3}, & \text{if } \eta(t)>0, \\ 1, & \text{if } \eta(t)=0,\ \mathcal{G}(t)>G_{l+1}, \\ 1, & \text{with probability } q \text{ if } \eta(t)=0,\ \mathcal{G}(t)=G_{l+1}, \\ 0, & \text{otherwise,} \end{cases} \qquad \text{(A.48)}$$

for $t = 1, 2, \ldots, T$. Policy $\Phi 3'$ may not be feasible since it continues to assign tasks to workers even when there are no such workers left. Note that $\{X_t^{\Phi 3}\}$ are not IID while $\{X_t^{\Phi 3'}\}$ are IID, for $t = 1, 2, \ldots, T$.

For a fixed T, define $U_s \triangleq \min\{k \in \mathbb{Z} : \sum_{t=1}^{k} X_t^{\Phi 3'} = \eta\}$ and $U_{ns} \triangleq \min\{k \in \mathbb{Z} : \sum_{t=1}^{k}(1 - X_t^{\Phi 3'}) = T - \eta\}$. Since $\{X_t^{\Phi 3'}\}$ are IID for $t = 1, 2, \ldots, T$ with $\mathbb{E}[X_t^{\Phi 3'}] = \mathbb{P}(X_t^{\Phi 3'} = 1) = \sum_{k=l+2}^{L} p_{\mathcal{G}}(G_k) + q p_{\mathcal{G}}(G_{l+1}) = 1 - \zeta, \mathbb{E}[1 - X_t^{\Phi 3'}] = \mathbb{P}(X_t^{\Phi 3'} = 0) = \zeta$, U_s is the sum of η IID geometric random variables with mean $1/(1-\zeta)$ and U_{ns} is the sum of $T - \eta$ IID geometric random variables with mean $1/\zeta$. Since $\eta = \lfloor T(1-\zeta) \rfloor \to +\infty$ and $T - \eta = \lceil T\zeta \rceil \to +\infty$ as $T \to +\infty$, then from the weak law of large numbers, $U_s/\eta \to 1/(1-\zeta)$ and $U_{ns}/(T-\eta) \to 1/\zeta$ in probability. Therefore, for any $\epsilon > 0$,

$$\begin{aligned}
\lim_{T\to+\infty} \mathbb{P}\left(\frac{U_s}{T} < 1 - \epsilon\right) &= \lim_{T\to+\infty} \mathbb{P}\left(\frac{U_s}{\eta} < \frac{1-\epsilon}{\lfloor T(1-\zeta)\rfloor/T}\right) \\
&\leq \lim_{T\to+\infty} \mathbb{P}\left(\frac{U_s}{\eta} < \frac{1-\epsilon}{1-\zeta} + \frac{1/T}{(1-\zeta)(1-\zeta-1/T)}\right) = 0, \\
\lim_{T\to+\infty} \mathbb{P}\left(\frac{U_{ns}}{T} < 1 - \epsilon\right) &= \lim_{T\to+\infty} \mathbb{P}\left(\frac{U_{ns}}{T-\eta} < \frac{1-\epsilon}{\lceil T\zeta\rceil/T}\right) \\
&\leq \lim_{T\to+\infty} \mathbb{P}\left(\frac{U_{ns}}{T-\eta} < \frac{1-\epsilon}{\zeta}\right) = 0.
\end{aligned}$$

Define $U_{min} \triangleq \min\{U_s, U_{ns}\}$. By definition, $U_{min}/T \leq 1$ and

$$\lim_{T\to+\infty} \mathbb{P}\left(\frac{U_{min}}{T} < 1 - \epsilon\right) \leq \lim_{T\to+\infty} \mathbb{P}\left(\frac{U_s}{T} < 1 - \epsilon\right) + \lim_{T\to+\infty} \mathbb{P}\left(\frac{U_{ns}}{T} < 1 - \epsilon\right) = 0. \tag{A.49}$$

Therefore, U_{min}/T converges to 1 in probability. Since U_{min}/T is bounded above by 1 and U_{min}/T converges to 1 in probability, U_{min}/T converges to 1 in mean (i.e., $\lim_{T\to+\infty} \mathbb{E}[U_{min}]/T = 1$).

The asymptotic expected weighted reward per task for WOSA-**w** under

policy $\Phi 3'$ is given by

$$\begin{aligned}
&\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\sum_{t=1}^{T}X_t^{\Phi 3}\mathcal{G}(t)]\\
&=\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\sum_{t=1}^{U_{min}}X_t^{\Phi 3}\mathcal{G}(t)+\mathbf{1}_{U_{min}<T}\sum_{t=U_{min}+1}^{T}X_t^{\Phi 3}\mathcal{G}(t)],\\
&=\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\sum_{t=1}^{U_{min}}X_t^{\Phi 3}\mathcal{G}(t)]+\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\mathbf{1}_{U_{min}<T}\sum_{t=U_{min}+1}^{T}X_t^{\Phi 3}\mathcal{G}(t)], \quad \text{(A.50)}
\end{aligned}$$

where $\mathbf{1}_{U_{min}<T}$ is the indicator function and when $U_{min}=T$ the second term in (A.50) becomes zero. Since U_{min} is a stopping time with respect to $\{X_t^{\Phi 3'}\}$ and $X_t^{\Phi 3}=X_t^{\Phi 3'}$ for $t=1,2,\ldots,U_{min}$, then by Wald's equation (See Appendix A.1), the first term in (A.50) is

$$\begin{aligned}
\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\sum_{t=1}^{U_{min}}X_t^{\Phi 3}\mathcal{G}(t)]&=\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\sum_{t=1}^{U_{min}}X_t^{\Phi 3'}\mathcal{G}(t)]\\
&=\lim_{T\to+\infty}\frac{\mathbb{E}[U_{min}]}{T}\mathbb{E}[X_t^{\Phi 3'}\mathcal{G}(t)]=\mathbb{E}[X_t^{\Phi 3'}\mathcal{G}(t)].
\end{aligned} \quad \text{(A.51)}$$

Consider the second term in (A.50): if $U_{min}<T$, then $X_t^{\Phi 3}\le X_t^{\Phi 3'}$ for $t=U_{min}+1,\ldots,T$, and hence

$$\begin{aligned}
&\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\mathbf{1}_{U_{min}<T}\sum_{t=U_{min}+1}^{T}X_t^{\Phi 3}\mathcal{G}(t)]\\
\le&\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\mathbf{1}_{U_{min}<T}\sum_{t=U_{min}+1}^{T}X_t^{\Phi 3'}\mathcal{G}(t)]\\
=&\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\sum_{t=1}^{T}X_t^{\Phi 3'}\mathcal{G}(t)-\sum_{t=1}^{U_{min}}X_t^{\Phi 3'}\mathcal{G}(t)]\\
\overset{(A.51)}{=}&\lim_{T\to+\infty}\left(1-\frac{\mathbb{E}[U_{min}]}{T}\right)\mathbb{E}[X_t^{\Phi 3'}\mathcal{G}(t)]=0.
\end{aligned} \quad \text{(A.52)}$$

Substituting (A.51) and (A.52) into (A.50),

$$\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\sum_{t=1}^{T}X_t^{\Phi 3}\mathcal{G}(t)]=\mathbb{E}[X_t^{\Phi 3'}\mathcal{G}(t)]=\left(\sum_{k=l+2}^{L}G_k p_{\mathcal{G}}(G_k)\right)+qG_{l+1}p_{\mathcal{G}}(G_{l+1}).$$

■

A.9 Proof of Theorem 7

Following the same arguments as in the proof of Theorem 6,

$$\begin{aligned}
&\lim_{T\to+\infty} r_j(\Phi 3)\\
=&\lim_{T\to+\infty}\frac{1}{T}\mathbb{E}[\sum_{t=1}^{U_{min}} X_t^{\Phi 3}\mathcal{A}_j(t) + \mathbf{1}_{U_{min}<T}\sum_{t=U_{min}+1}^{T} X_t^{\Phi 3}\mathcal{A}_j(t)],\\
\overset{(a)}{=}&\mathbb{E}[X_t^{\Phi 3'}\mathcal{A}_j(t)]\\
=&\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)>G_{l+1}]\mathbb{P}_{\mathcal{G}}(\mathcal{G}(t)>G_{l+1}) + q\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_{l+1}]p_{\mathcal{G}}(G_{l+1}),
\end{aligned}$$

where q is given by (2.12) and equality (a) follows from the same arguments as (A.50), (A.51) and (A.52) with $\mathcal{G}(t)$ substituted by $\mathcal{A}_j(t)$ (see Appendix A.8). This expression is the same as $\rho_j^{\zeta}(\mathbf{w})$ given by (2.13). ■

A.10 Proof of Lemma 2

From Theorem 1, $\mathbb{E}[\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}] = a_{\lceil T\theta\rceil,0}$. Therefore,

$$\begin{aligned}
&|a_{\lceil T\theta\rceil,0} - G_{l+1}| = |\mathbb{E}[\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}] - G_{l+1}|\\
=&|\mathbb{E}[\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}|\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} = G_{l+1}]\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} = G_{l+1})\\
&+\mathbb{E}[\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}|\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}]\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}) - G_{l+1}|\\
=&|\mathbb{E}[\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}|\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}] - G_{l+1}|\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1})\\
\leq&G_L\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}).
\end{aligned} \tag{A.53}$$

We need to show $\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}) \to 0$ with an exponential rate.

Since $\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}$ assumes discrete values, there exists $\epsilon > 0$ small such that

$$\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} \neq G_{l+1}) = \mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} > G_{l+1}+\epsilon) + \mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} < G_{l+1}-\epsilon).$$

Since $\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)}$ is monotonically increasing with respect to θ, if $\hat{\mathcal{G}}_T^{(\lceil T\theta\rceil)} > G_{l+1}+\epsilon$, then $\hat{\mathcal{G}}_T^{(\lceil T\theta'\rceil)} > G_{l+1}+\epsilon$, for any $\theta \leq \theta' < 1$. Define $Y_t^G \triangleq \mathbf{1}_{\mathcal{G}(t)>G_{l+1}+\epsilon}$

(indicator function), with $\mathbb{E}[Y_t^G] = \mathbb{P}(\mathcal{G}(t) > G_{l+1} + \epsilon) = 1 - F_{\mathcal{G}}(G_{l+1})$. Then,

$$\begin{aligned}
\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta \rceil)} > G_{l+1} + \epsilon) &= \mathbb{P}\left(\sum_{t=1}^{T} \mathbf{1}_{\mathcal{G}(t) > G_{l+1}+\epsilon} > T - \lceil T\theta \rceil + 1\right) \\
&\leq \mathbb{P}\left(\sum_{t=1}^{T} Y_t^G > T - T\theta\right) \\
&= \mathbb{P}\left(\sum_{t=1}^{T} Y_t^G - \sum_{t=1}^{T} \mathbb{E}[Y_t^G] > T(F_{\mathcal{G}}(G_{l+1}) - \theta)\right) \\
&\leq \exp(-2T(F_{\mathcal{G}}(G_{l+1}) - \theta)^2), \qquad \text{(A.54)}
\end{aligned}$$

where the last inequality follows from Hoeffding inequality [73].

Similarly, if $\hat{\mathcal{G}}_T^{(\lceil T\theta \rceil)} < G_{l+1} - \epsilon$, then $\hat{\mathcal{G}}_T^{(\lceil T\theta' \rceil)} < G_{l+1} - \epsilon$, for any $0 < \theta' \leq \theta$. Define $Z_t^G \triangleq \mathbf{1}_{\mathcal{G}(t) < G_{l+1} - \epsilon}$, with $\mathbb{E}[Z_t^G] = \mathbb{P}(\mathcal{G}(t) < G_{l+1} - \epsilon) = F_{\mathcal{G}}(G_l)$, and hence

$$\begin{aligned}
\mathbb{P}(\hat{\mathcal{G}}_T^{(\lceil T\theta \rceil)} < G_{l+1} - \epsilon) &= \mathbb{P}\left(\sum_{t=1}^{T} \mathbf{1}_{\mathcal{G}(t) < G_{l+1}-\epsilon} > \lceil T\theta \rceil\right) \\
&\leq \mathbb{P}\left(\sum_{t=1}^{T} Z_t^G > T\theta\right) \\
&= \mathbb{P}\left(\sum_{t=1}^{T} Z_t^G - \sum_{t=1}^{T} \mathbb{E}[Z_t^G] > T(\theta - F_{\mathcal{G}}(G_l))\right) \\
&\leq \exp(-2T(\theta - F_{\mathcal{G}}(G_l))^2), \qquad \text{(A.55)}
\end{aligned}$$

where the last inequality follows from Hoeffding inequality [73].

Substituting (A.54) and (A.55) into (A.53),

$$|a_{\lceil T\theta \rceil, 0} - G_{l+1}| \leq 2G_L \exp(-2T\Delta_\theta^2),$$

where

$$\Delta_\theta = \min\{F_{\mathcal{G}}(G_{l+1}) - \theta, \theta - F_{\mathcal{G}}(G_l)\}. \qquad \blacksquare$$

A.11 Proof of Theorem 8

From Corollary 1,

$$|R_w(\Phi 1) - \rho_w^\zeta(\mathbf{w})| = |\frac{1}{T}\mathbb{E}[\sum_{t=1}^{T} X_t^{\Phi 1}\mathcal{G}(t)] - \rho_w^\zeta(\mathbf{w})| = |\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T} a_{i,0} - \rho_w^\zeta(\mathbf{w})|. \tag{A.56}$$

For any $\epsilon > 0$ small, use the partition defined in the proof of Theorem 3, i.e.,

$$I \triangleq [\lceil T\zeta\rceil + 1, T] = I_e \bigcup \left(\bigcup_{m=l+1,\ldots,L} I_m(2\epsilon, 2\epsilon) \right),$$

where $I_m(2\epsilon, 2\epsilon)$ for $m = l+1, \ldots, L$ and I_e are defined by (A.24) and (A.25) with $\epsilon_1 = \epsilon_2 = 2\epsilon$. Then,

$$\begin{aligned}
\rho_w^\zeta(\mathbf{w}) &= \left(\sum_{m=l+2}^{L} G_m p_{\mathcal{G}}(G_m) \right) + qG_{l+1}p_{\mathcal{G}}(G_{l+1}) \\
&= \frac{1}{T}\left(\left(\sum_{m=l+2}^{L} G_m T p_{\mathcal{G}}(G_m) \right) + qG_{l+1}Tp_{\mathcal{G}}(G_{l+1}) \right) \\
&= \frac{1}{T}\sum_{i=1}^{T}\sum_{m=l+1}^{L} \left(G_m \mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} + G_{m_i}\mathbf{1}_{i\in I_e} \right),
\end{aligned}$$

where $G_{m_i} = \arg\min_{G_m,\ m=1,2,\ldots,L} F_{\mathcal{G}}(G_m) \geq i/T$. Therefore,

$$\begin{aligned}
&|\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T} a_{i,0} - \rho_w^\zeta(\mathbf{w})| \\
=&\frac{1}{T}|\sum_{i=\lceil T\zeta\rceil+1}^{T} a_{i,0} - \sum_{i=1}^{T}\sum_{m=l+1}^{L} \left(G_m \mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} + G_{m_i}\mathbf{1}_{i\in I_e} \right)| \\
=&\frac{1}{T}|\sum_{i=\lceil T\zeta\rceil+1}^{T}\sum_{m=l+1}^{L} (a_{i,0} - G_m)\mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} + \sum_{i=\lceil T\zeta\rceil+1}^{T} (a_{i,0} - G_{m_i})\mathbf{1}_{i\in I_e}| \\
\leq&\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}\sum_{m=l+1}^{L} |a_{i,0} - G_m|\mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} + \frac{\mu(I_e)}{T}G_L.
\end{aligned} \tag{A.57}$$

Moreover, from the definition of I_e in (A.25),

$$\begin{aligned}\mu(I_e) = &\sum_{m=l+1}^{L-1} \left(\lfloor T(F_{\mathcal{G}}(G_m) + 2\epsilon)\rfloor - \lceil T(F_{\mathcal{G}}(G_m) - 2\epsilon)\rceil\right) \\ &+ \left(\lfloor T(\zeta + 2\epsilon)\rfloor - \lceil T\zeta\rceil\right) + \left(T - \lceil T(F_{\mathcal{G}}(G_L) - 2\epsilon)\rceil\right) \\ \leq &4(L-l)T\epsilon, \end{aligned} \tag{A.58}$$

and hence the second term in (A.57) becomes

$$\frac{\mu(I_e)}{T} G_L \leq 4(L-l)G_L\epsilon. \tag{A.59}$$

For the first term in (A.57), note that for any $T > \lceil 2/\epsilon\rceil$,

$$\begin{aligned}\lfloor T(F_{\mathcal{G}}(G_m) + 2\epsilon)\rfloor &\geq T(F_{\mathcal{G}}(G_m) + 2\epsilon) - 1 \geq T(F_{\mathcal{G}}(G_m) + \epsilon), \\ \lceil T(F_{\mathcal{G}}(G_m) - 2\epsilon)\rceil + 1 &\leq T(F_{\mathcal{G}}(G_m) - 2\epsilon) + 2 \leq T(F_{\mathcal{G}}(G_m) - \epsilon),\end{aligned}$$

for $m = l+1, \ldots, L$. Then for any $T > \lceil 2/\epsilon\rceil$, from Lemma 2,

$$|a_{i,0} - G_m| \leq 2G_L \exp(-2T\Delta_{i/T}^2),$$

where $\Delta_{i/T} = \min\{F_{\mathcal{G}}(G_m) - i/T, i/T - F_{\mathcal{G}}(G_{m-1})\}$ for $i \in I_m(2\epsilon, 2\epsilon)$ and $m = l+1, \ldots, L$. Therefore,

$$\begin{aligned}&\frac{1}{T} \sum_{i=\lceil T\zeta\rceil+1}^{T} \sum_{m=l+1}^{L} |a_{i,0} - G_m| \mathbf{1}_{i \in I_m(2\epsilon,2\epsilon)} \\ \leq &\frac{1}{T} \sum_{m=l+1}^{L} 2G_L \sum_{i=\lceil T\zeta\rceil+1}^{T} \exp(-2T\Delta_{i/T}^2) \mathbf{1}_{i \in I_m(2\epsilon,2\epsilon)} \\ \leq &\frac{2G_L}{T} \sum_{m=l+1}^{L} \sum_{i=\lfloor T(F_{\mathcal{G}}(G_{m-1})+2\epsilon)\rfloor+1}^{\lceil T(F_{\mathcal{G}}(G_m)-2\epsilon)\rceil} \exp(-2T\Delta_{i/T}^2). \end{aligned} \tag{A.60}$$

For each $m = l+1, \ldots, L$, we have

$$\begin{aligned}
&\sum_{i=\lfloor T(F_{\mathcal{G}}(G_{m-1})+2\epsilon)\rfloor+1}^{\lceil T(F_{\mathcal{G}}(G_m)-2\epsilon)\rceil} \exp(-2T\Delta_{i/T}^2) \\
=&2\sum_{i=\lceil \frac{T}{2}(F_{\mathcal{G}}(G_{m-1})+F_{\mathcal{G}}(G_m))\rceil}^{\lceil T(F_{\mathcal{G}}(G_m)-2\epsilon)\rceil} \exp(-2T\Delta_{i/T}^2) \\
=&2\sum_{i=\lceil \frac{T}{2}(F_{\mathcal{G}}(G_{m-1})+F_{\mathcal{G}}(G_m))\rceil}^{\lceil T(F_{\mathcal{G}}(G_m)-2\epsilon)\rceil} \exp(-2T(F_{\mathcal{G}}(G_m)-\frac{i}{T})^2) \\
\overset{(a)}{\leq}&2\int_{\lceil \frac{T}{2}(F_{\mathcal{G}}(G_{m-1})+F_{\mathcal{G}}(G_m))\rceil}^{\lceil T(F_{\mathcal{G}}(G_m)-2\epsilon)\rceil+1} \exp\left(-2T(F_{\mathcal{G}}(G_m)-\frac{i}{T})^2\right) di \\
\leq&2\int_{\frac{T}{2}(F_{\mathcal{G}}(G_{m-1})+F_{\mathcal{G}}(G_m))}^{T(F_{\mathcal{G}}(G_m)-\epsilon)} \exp\left(-2T(F_{\mathcal{G}}(G_m)-\frac{i}{T})^2\right) di \\
\overset{(b)}{=}&\sqrt{T}\int_{2\sqrt{T}\epsilon}^{\sqrt{T}(F_{\mathcal{G}}(G_m)-F_{\mathcal{G}}(G_{m-1})} \exp(-\frac{y^2}{2})dy \\
=&\frac{\sqrt{T}}{\sqrt{2\pi}}\left(Q(2\sqrt{T}\epsilon)-Q(\sqrt{T}(F_{\mathcal{G}}(G_m)-F_{\mathcal{G}}(G_{m-1})))\right), \\
\leq&\frac{\sqrt{T}}{2\sqrt{2\pi}},
\end{aligned} \tag{A.61}$$

where Q is the tail probability of the standard normal distribution. Inequality (a) follows from the fact that $\exp\left(-2T(F_{\mathcal{G}}(G_m)-\frac{i}{T})^2\right)$ is an increasing function of $i \in \left[\lceil \frac{T}{2}(F_{\mathcal{G}}(G_{m-1})+F_{\mathcal{G}}(G_m))\rceil, \lceil T(F_{\mathcal{G}}(G_m)-2\epsilon)\rceil+1\right]$, and inequality (b) follows from a change of variable $y = 2\sqrt{T}(F_{\mathcal{G}}(G_m)-\frac{i}{T})$. Moreover, since $T > \lceil 2/\epsilon \rceil$, setting $\epsilon = 3/T = O(1/T)$, then substituting (A.61) into (A.60), we have

$$\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}\sum_{m=l+1}^{L}|a_{i,0}-G_m|\mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} \leq \frac{2G_L}{T}(L-l)\frac{\sqrt{T}}{2\sqrt{2\pi}} = O(\frac{1}{\sqrt{T}}). \tag{A.62}$$

Substituting (A.59) and (A.62) into (A.57),

$$|\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T} a_{i,0} - \rho_w^{\zeta}(\mathbf{w})| \leq O(\frac{1}{\sqrt{T}}) + O(\frac{1}{T}) = O(\frac{1}{\sqrt{T}}).$$

Therefore, $R_w(\Phi 1)$ converges to $\rho_w^{\zeta}(\mathbf{w})$ with rate $O(1/\sqrt{T})$. ■

A.12 Proof of Theorem 9

For $j = 1, 2, \ldots, n$, from (2.10),

$$|r_j(\Phi 1) - \rho_j^{\zeta}(\mathbf{w})| = |\frac{1}{T}\mathbb{E}[\sum_{t=1}^{T} X_t^{\Phi 1}\mathcal{A}_j(t)] - \rho_j^{\zeta}(\mathbf{w})| = |\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T} b_{i,0}^j - \rho_j^{\zeta}(\mathbf{w})|. \tag{A.63}$$

The proof of the convergence rate of $r_j(\Phi 1)$ follows the same arguments as in the proof of Theorem 8. Here, we mainly provide the key steps. Use the same partition as in the proof of Theorem 8, i.e.,

$$I \triangleq [\lceil T\zeta\rceil + 1, T] = I_e \bigcup \left(\bigcup_{m=l+1,\ldots,L} I_m(2\epsilon, 2\epsilon) \right).$$

Note that

$$\rho_j^{\zeta}(\mathbf{w}) = \frac{1}{T}\sum_{i=1}^{T}\sum_{m=l+1}^{L} \left(\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_m]\mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} + \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_{m_i}]\mathbf{1}_{i\in I_e} \right),$$

where $G_{m_i} = \arg\min_{G_m,\ m=1,2,\ldots,L} F_{\mathcal{G}}(G_m) \geq i/T$. Therefore,

$$\begin{aligned}
&|\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T} b_{i,0}^j - \rho_j^{\zeta}(\mathbf{w})| \\
=&\frac{1}{T}|\sum_{i=\lceil T\zeta\rceil+1}^{T} b_{i,0}^j \\
&- \sum_{i=1}^{T}\sum_{m=l+1}^{L} \left(\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_m]\mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} + \mathbb{E}[\mathcal{A}_j(t)vert\mathcal{G}(t) = G_{m_i}]\mathbf{1}_{i\in I_e} \right)| \\
\leq&\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}\sum_{m=l+1}^{L} |b_{i,0}^j - \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_m]|\mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} + \frac{\mu(I_e)}{T}A_M. \quad \text{(A.64)}
\end{aligned}$$

Similarly, from (A.58) the second term in (A.64) becomes

$$\frac{\mu(I_e)}{T}A_M \leq 4(L-l)A_M\epsilon. \tag{A.65}$$

For the first term in (A.64), for any $T > \lceil 2/\epsilon \rceil$ from Lemma 3,

$$|b_{i,0}^j - \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t) = G_m]| \leq 4A_M \exp(-2T\Delta_{i/T}^2),$$

where $\Delta_{i/T} = \min\{F_{\mathcal{G}}(G_m) - i/T, i/T - F_{\mathcal{G}}(G_{m-1})\}$ for $i \in I_m(2\epsilon, 2\epsilon)$ and $m = l+1, \ldots, L$. Therefore,

$$\begin{aligned}
&\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}\sum_{m=l+1}^{L}|b_{i,0}^{j} - \mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_m]|\mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} \\
\leq &\frac{1}{T}\sum_{m=l+1}^{L}4A_M\sum_{i=\lceil T\zeta\rceil+1}^{T}\exp(-2T\Delta_{i/T}^2)\mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} \\
\leq &\frac{4A_M}{T}\sum_{m=l+1}^{L}\sum_{i=\lfloor T(F_{\mathcal{G}}(G_{m-1})+2\epsilon)\rfloor+1}^{\lceil T(F_{\mathcal{G}}(G_m)-2\epsilon)\rceil}\exp(-2T\Delta_{i/T}^2). \qquad \text{(A.66)}
\end{aligned}$$

Moreover, since $T > \lceil 2/\epsilon\rceil$, set $\epsilon = 3/T = O(1/T)$, then substituting (A.61) into (A.66),

$$\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}\sum_{m=l+1}^{L}|b_{i,0}^{j}-\mathbb{E}[\mathcal{A}_j(t)|\mathcal{G}(t)=G_m]|\mathbf{1}_{i\in I_m(2\epsilon,2\epsilon)} \leq \frac{4A_M}{T}(L-l)\frac{\sqrt{T}}{2\sqrt{2\pi}} = O(\frac{1}{\sqrt{T}}). \qquad \text{(A.67)}$$

Substituting (A.65) and (A.67) into (A.64),

$$|\frac{1}{T}\sum_{i=\lceil T\zeta\rceil+1}^{T}b_{i,0}^{j} - \rho_j^{\zeta}(\mathbf{w})| \leq O(\frac{1}{\sqrt{T}}) + O(\frac{1}{T}) = O(\frac{1}{\sqrt{T}}).$$

Therefore, $r_j(\Phi 1)$ converges to $\rho_j^{\zeta}(\mathbf{w})$ with rate $O(1/\sqrt{T})$. ■

A.13 Proof of Theorem 10

Following the same arguments as in the proof of Theorem 4, we break $R_w(\Phi 2)$ into two parts as follows:

$$\begin{aligned} R_w(\Phi 2) &= \frac{1}{T}\mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}\mathcal{G}(t) + \sum_{t=\lfloor \nu_\zeta T \rfloor+1}^{T} X_t^{\Phi 2}\mathcal{G}(t)] \\ &= \frac{1}{T}\mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}\mathcal{G}(t)] + \frac{1}{T}\mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor+1}^{T} X_t^{\Phi 1}\mathcal{G}(t)] \\ &= \frac{1}{T}\lfloor \nu_\zeta T \rfloor \rho_w^\zeta(w) + \frac{1}{T}\mathbb{E}\left[\mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor+1}^{T} X_t^{\Phi 1}\mathcal{G}(t)|\zeta(\lfloor \nu_\zeta T \rfloor)]\right], \quad \text{(A.68)} \end{aligned}$$

where the last line follows from $\{X_t^{\Phi 2}\mathcal{G}(t)\}$ being IID for $t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor$ and $\zeta(\lfloor \nu_\zeta T \rfloor)$ given by (A.35) is a random variable. For the second term in (A.68), the outer and inner expectations are taken with respect to $\zeta(\lfloor \nu_\zeta T \rfloor)$ and $\{\mathcal{G}(t)\}_{t=\lfloor \nu_\zeta T \rfloor+1}^{T}$, respectively. Therefore,

$$\begin{aligned} &|R_w(\Phi 2) - \rho_w^\zeta(\mathbf{w})| \\ =&\frac{T - \lfloor \nu_\zeta T \rfloor}{T}|\frac{1}{T - \lfloor \nu_\zeta T \rfloor}\mathbb{E}\left[\mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor+1}^{T} X_t^{\Phi 1}\mathcal{G}(t)|\zeta(\lfloor \nu_\zeta T \rfloor)]\right] - \rho_w^\zeta(\mathbf{w})| \\ \leq&|\mathbb{E}\left[\frac{1}{T - \lfloor \nu_\zeta T \rfloor}\mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor+1}^{T} X_t^{\Phi 1}\mathcal{G}(t)|\zeta(\lfloor \nu_\zeta T \rfloor)] - \rho_w^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})\right]| \\ &+ |\mathbb{E}[\rho_w^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})] - \rho_w^\zeta(\mathbf{w})|. \quad \text{(A.69)} \end{aligned}$$

From (A.36) and (A.37), $F_{\mathcal{G}}(G_l) \leq \zeta(\lfloor \nu_\zeta T \rfloor) \leq F_{\mathcal{G}}(G_{l+1})$ holds. Therefore, from Theorem 8,

$$|\frac{1}{T - \lfloor \nu_\zeta T \rfloor}\mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor+1}^{T} X_t^{\Phi 1}\mathcal{G}(t)|\zeta(\lfloor \nu_\zeta T \rfloor)] - \rho_w^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})| = O(\frac{1}{\sqrt{T}}), \quad \text{(A.70)}$$

for any $\zeta(\lfloor \nu_\zeta T \rfloor)$. Therefore, the first term in (A.69) is $O(1/\sqrt{T})$. For the second term in (A.69), since $\rho_w^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})$ is an affine function of $\zeta(\lfloor \nu_\zeta T \rfloor)$,

then from Theorem 2,

$$|\mathbb{E}[\rho_w^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})] - \rho_w^\zeta(\mathbf{w})| = |\mathbb{E}[\zeta(\lfloor \nu_\zeta T \rfloor)] - \zeta| p_{\mathcal{G}}(G_{l+1}) G_{l+1}.$$

Recall that from(A.41),

$$\mathbb{E}[\zeta(\lfloor \nu_\zeta T \rfloor)] = 1 - \frac{\lfloor (1-\zeta)T \rfloor - \mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}]}{T - \lfloor \nu_\zeta T \rfloor} = 1 - \frac{\lfloor (1-\zeta)T \rfloor - (1-\zeta)\lfloor \nu_\zeta T \rfloor}{T - \lfloor \nu_\zeta T \rfloor},$$

where the last equality follows from $X_t^{\Phi 2}$ being IID for $t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor$, with $\mathbb{E}[X_t^{\Phi 2}] = 1 - \zeta$. Therefore,

$$|\mathbb{E}[\zeta(\lfloor \nu_\zeta T \rfloor)] - \zeta| = \frac{(1-\zeta)T - \lfloor (1-\zeta)T \rfloor}{T - \lfloor \nu_\zeta T \rfloor} = O(\frac{1}{T}). \tag{A.71}$$

Substituting (A.70) and (A.71) into (A.69), we have

$$|R_w(\Phi 2) - \rho_w^\zeta(\mathbf{w})| \leq O(\frac{1}{\sqrt{T}}) + O(\frac{1}{T}) = O(\frac{1}{\sqrt{T}}).$$

For $r_j(\Phi 2)$, we use the same technique:

$$\begin{aligned} r_j(\Phi 2) &= \frac{1}{T}\mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}\mathcal{A}_j(t) + \sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 2}\mathcal{A}_j(t)] \\ &= \frac{1}{T}\mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} X_t^{\Phi 2}\mathcal{A}_j(t)] + \frac{1}{T}\mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 1}\mathcal{A}_j(t)] \\ &= \frac{1}{T}\sum_{k=l+1}^{L}\mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} \Gamma_t^{(k)}\mathcal{A}_j(t)] + \frac{1}{T}\mathbb{E}\left[\mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 1}\mathcal{A}_j(t) | \zeta(\lfloor \nu_\zeta T \rfloor)]\right], \end{aligned} \tag{A.72}$$

where $\{\Gamma_t^{(k)}\mathcal{A}_j(t)\}$ for each fixed $k = l+1, \ldots, L$ and $j = 1, 2, \ldots, n$ are IID for $t = 1, 2, \ldots, \lfloor \nu_\zeta T \rfloor$, with mean values given by (A.44) and $\zeta(\lfloor \nu_\zeta T \rfloor)$ given by (A.35) is a random variable. For the second term in (A.72), the outer and inner expectations are taken with respect to $\zeta(\lfloor \nu_\zeta T \rfloor)$ and $\{\mathcal{A}_j(t)\}_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T}$, respectively. Therefore,

$$\frac{1}{T}\sum_{k=l+1}^{L}\mathbb{E}[\sum_{t=1}^{\lfloor \nu_\zeta T \rfloor} \Gamma_t^{(k)}\mathcal{A}_j(t)] = \frac{\lfloor \nu_\zeta T \rfloor}{T}\sum_{k=l+1}^{L}\mathbb{E}[\Gamma_t^{(k)}\mathcal{A}_j(t)] = \frac{\lfloor \nu_\zeta T \rfloor}{T}\rho_j^\zeta(\mathbf{w}),$$

and

$$
\begin{aligned}
&|r_j(\Phi 2) - \rho_j^{\zeta}(\mathbf{w})| \\
=&\frac{T - \lfloor \nu_\zeta T \rfloor}{T} |\frac{1}{T - \lfloor \nu_\zeta T \rfloor} \mathbb{E}\left[\mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 1} \mathcal{A}_j(t) | \zeta(\lfloor \nu_\zeta T \rfloor)] \right] - \rho_j^{\zeta}(\mathbf{w})| \\
\leq &|\mathbb{E}\left[\frac{1}{T - \lfloor \nu_\zeta T \rfloor} \mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 1} \mathcal{A}_j(t) | \zeta(\lfloor \nu_\zeta T \rfloor)] - \rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w}) \right]| \\
&+ |\mathbb{E}[\rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})] - \rho_j^{\zeta}(\mathbf{w})|. \qquad \text{(A.73)}
\end{aligned}
$$

From (A.36) and (A.37), $F_{\mathcal{G}}(G_l) \leq \zeta(\lfloor \nu_\zeta T \rfloor) \leq F_{\mathcal{G}}(G_{l+1})$ holds. Therefore, from Theorem 9,

$$
|\frac{1}{T - \lfloor \nu_\zeta T \rfloor} \mathbb{E}[\sum_{t=\lfloor \nu_\zeta T \rfloor + 1}^{T} X_t^{\Phi 1} \mathcal{A}_j(t) | \zeta(\lfloor \nu_\zeta T \rfloor)] - \rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})| = O(\frac{1}{\sqrt{T}}), \quad \text{(A.74)}
$$

for any $\zeta(\lfloor \nu_\zeta T \rfloor)$. Therefore, the first term in (A.73) is $O(1/\sqrt{T})$. For the second term in (A.73), since $\rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})$ is an affine function of $\zeta(\lfloor \nu_\zeta T \rfloor)$, then from Theorem 3,

$$
|\mathbb{E}[\rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})] - \rho_j^{\zeta}(\mathbf{w})| = |\mathbb{E}[\zeta(\lfloor \nu_\zeta T \rfloor)] - \zeta| p_{\mathcal{G}}(G_{l+1}) \mathbb{E}[\mathcal{A}_j(t) | \mathcal{G}(t) = G_{l+1}].
$$

Therefore, from (A.71),

$$
\begin{aligned}
|\mathbb{E}[\rho_j^{\zeta(\lfloor \nu_\zeta T \rfloor)}(\mathbf{w})] - \rho_j^{\zeta}(\mathbf{w})| &= p_{\mathcal{G}}(G_{l+1}) \mathbb{E}[\mathcal{A}_j(t) | \mathcal{G}(t) = G_{l+1}] \frac{(1-\zeta)T - \lfloor (1-\zeta)T \rfloor}{T - \lfloor \nu_\zeta T \rfloor} \\
&= O(\frac{1}{T}). \qquad \text{(A.75)}
\end{aligned}
$$

Substituting (A.74) and (A.75) into (A.73), we have

$$
|r_j(\Phi 2) - \rho_j^{\zeta}(\mathbf{w})| \leq O(\frac{1}{\sqrt{T}}) + O(\frac{1}{T}) = O(\frac{1}{\sqrt{T}}).
$$

■

A.14 Proof of Lemma 4

We use the fact that U_s is the sum of η IID geometric random variables with mean $1/(1-\zeta)$ and U_{ns} is the sum of $T-\eta$ IID geometric random variables with mean $1/\zeta$. Moreover, for any $\epsilon > 0$,

$$\mathbb{P}\left(\frac{U_{min}}{T} < 1-\epsilon\right) \leq \mathbb{P}\left(\frac{U_s}{T} < 1-\epsilon\right) + \mathbb{P}\left(\frac{U_{ns}}{T} < 1-\epsilon\right).$$

Therefore, we prove Lemma 4 by showing $\mathbb{P}\left(\frac{U_s}{T} < 1-\epsilon\right)$ and $\mathbb{P}\left(\frac{U_{ns}}{T} < 1-\epsilon\right)$ converge to 0 with exponential rates as $T \to +\infty$ using the large deviation theory.

Consider U_s/T first. Then

$$\mathbb{P}\left(\frac{U_s}{T} < 1-\epsilon\right) = \mathbb{P}\left(\frac{U_s}{\eta} < \frac{1-\epsilon}{\eta/T}\right) \leq \mathbb{P}\left(\frac{U_s}{\eta} < \frac{1-\epsilon}{1-\zeta} + \frac{1/T}{(1-\zeta)(1-\zeta-1/T)}\right).$$

There exists $T_N = \lceil \frac{2/\epsilon+1}{1-\zeta} \rceil \in \mathbb{Z}^+$ such that $\frac{1/T}{1-\zeta-1/T} < \epsilon/2$ for $T > T_N$. Therefore, for $T > T_N$,

$$\mathbb{P}\left(\frac{U_s}{T} < 1-\epsilon\right) \leq \mathbb{P}\left(\frac{U_s}{\eta} < \frac{1-\epsilon/2}{1-\zeta}\right) \leq \exp(-\eta l(\epsilon_1)), \tag{A.76}$$

where $\epsilon_1 = (1-\epsilon/2)/(1-\zeta)$ and the rate function is given by

$$l(y) = \sup_{\vartheta > 0} \left(\vartheta y - \ln \mathbb{E}[\exp(\vartheta \hat{Y})]\right), \tag{A.77}$$

with $\hat{Y}$ being the geometric random variable with mean $1/(1-\zeta)$. Therefore, substituting the moment generating function of geometric random variables into (A.77), we have

$$\begin{aligned} l(\epsilon_1) &= \sup_{\vartheta > 0} \left(\vartheta\epsilon_1 - (\ln(1-\zeta) + \vartheta - \ln(1-\zeta e^{\vartheta}))\right) \\ &= \frac{1}{1-\zeta}\left((\zeta - \epsilon/2)\ln(1-\frac{\epsilon}{2\zeta}) - (1-\epsilon/2)\ln(1-\epsilon/2)\right) > 0. \end{aligned} \tag{A.78}$$

Since $\eta = \lfloor T(1-\zeta) \rfloor \geq T(1-\zeta) - 1$, then substituting (A.78) into (A.76) we have

$$\begin{aligned}&\mathbb{P}\left(\frac{U_s}{T} < 1-\epsilon\right)\\ &\leq \exp\left(-(T - \frac{1}{1-\zeta})\left((\zeta - \epsilon/2)\ln(1 - \frac{\epsilon}{2\zeta}) - (1-\epsilon/2)\ln(1-\epsilon/2)\right)\right).\end{aligned} \tag{A.79}$$

For ϵ small, we use Taylor expansion to approximate the rate function. Since $\ln(1-x) \approx -x$,

$$\begin{aligned}&\mathbb{P}\left(\frac{U_s}{T} < 1-\epsilon\right)\\ &\leq \exp\left(-(T - \frac{1}{1-\zeta})\left((\frac{\epsilon}{2})^2\frac{1-\zeta}{\zeta}\right)\right) \leq e\exp\left(-T\left((\frac{\epsilon}{2})^2\frac{1-\zeta}{\zeta}\right)\right),\end{aligned}$$

for $\epsilon \leq 2\sqrt{\zeta}$. Similarly, for U_{ns}/T we have

$$\begin{aligned}\mathbb{P}\left(\frac{U_{ns}}{T} < 1-\epsilon\right) &= \mathbb{P}\left(\frac{U_{ns}}{T-\eta} < \frac{1-\epsilon}{T-\eta}\right)\\ &\leq \mathbb{P}\left(\frac{U_{ns}}{T-\eta} < \frac{1-\epsilon}{\zeta}\right) \leq \exp\left(-(T-\eta)l'(\epsilon)\right),\end{aligned}$$

with the rate function given by

$$l'(y) = \sup_{\vartheta > 0}\left(\vartheta y - \ln \mathbb{E}[\exp(\vartheta \hat{Y}')]\right),$$

with $\hat{Y}'$ being the geometric random variable with mean $1/\zeta$. Following the same arguments, since $T - \eta = \lceil T\zeta \rceil \geq T\zeta$, we have

$$\mathbb{P}\left(\frac{U_{ns}}{T} < 1-\epsilon\right) \leq \exp\left(-T\left((1-\zeta-\epsilon)\ln(1 - \frac{\epsilon}{1-\zeta}) - (1-\epsilon)\ln(1-\epsilon)\right)\right), \tag{A.80}$$

with $(1-\zeta-\epsilon)\ln(1-\epsilon/(1-\zeta)) - (1-\epsilon)\ln(1-\epsilon) > 0$. Using Taylor expansion to approximate rate function for ϵ small, we have

$$\mathbb{P}\left(\frac{U_{ns}}{T} < 1-\epsilon\right) \leq \exp\left(-T\left(\epsilon^2\frac{\zeta}{1-\zeta}\right)\right).$$

Combining (A.79) and (A.80) together, for $T > T_N$ and $\epsilon \leq 2\sqrt{\zeta}$,

$$\mathbb{P}\left(\frac{U_{min}}{T} < 1-\epsilon\right) < (1+e)\exp(-T\Delta_U^2),$$

where

$$\Delta_U = \sqrt{\min\{(\frac{\epsilon}{2})^2\frac{1-\zeta}{\zeta}, \epsilon^2\frac{\zeta}{1-\zeta}\}} = \epsilon\sqrt{\min\{\frac{1-\zeta}{4\zeta}, \frac{\zeta}{1-\zeta}\}},$$

which completes the proof. ■

A.15 Proof of Theorem 11

Following the arguments for (A.50),

$$\begin{aligned}|\frac{1}{T}\mathbb{E}[\sum_{t=1}^{T} X_t^{\Phi 3}\mathcal{G}(t)] - \rho_w^\zeta(\mathbf{w})| \leq & |\frac{1}{T}\mathbb{E}[\sum_{t=1}^{U_{min}} X_t^{\Phi 3}\mathcal{G}(t)] - \rho_w^\zeta(\mathbf{w})| \\ & + \frac{1}{T}\mathbb{E}[\mathbf{1}_{U_{min}<T}\sum_{t=U_{min}+1}^{T} X_t^{\Phi 3}\mathcal{G}(t)]. \quad \text{(A.81)}\end{aligned}$$

Note that $\{X_t^{\Phi 3}\mathcal{G}(t)\}$ are IID for $t = 1, 2, \ldots, U_{min}$, with $\mathbb{E}[X_t^{\Phi 3}\mathcal{G}(t)|t \leq U_{min}] = \rho_w^\zeta(\mathbf{w})$. Therefore,

$$\begin{aligned}|\frac{1}{T}\mathbb{E}[\sum_{t=1}^{U_{min}} X_t^{\Phi 3}\mathcal{G}(t)] - \rho_w^\zeta(\mathbf{w})| &= |\frac{1}{T}\mathbb{E}[U_{min}]\mathbb{E}[X_t^{\Phi 3}\mathcal{G}(t)] - \rho_w^\zeta(\mathbf{w})| \\ &= \left(1 - \frac{\mathbb{E}[U_{min}]}{T}\right)\rho_w^\zeta(\mathbf{w}), \quad \text{(A.82)}\end{aligned}$$

where the first equality follows from Wald's equation (See Appendix A.1). Following the arguments for (A.52),

$$\begin{aligned}\frac{1}{T}\mathbb{E}[\mathbf{1}_{U_{min}<T}\sum_{t=U_{min}+1}^{T}X_t^{\Phi 3}\mathcal{G}(t)] &\leq \frac{1}{T}\mathbb{E}[\mathbf{1}_{U_{min}<T}\sum_{t=U_{min}+1}^{T}X_t^{\Phi 3'}\mathcal{G}(t)] \\ &= \frac{1}{T}\mathbb{E}[\sum_{t=1}^{T}X_t^{\Phi 3'}\mathcal{G}(t)-\sum_{t=1}^{U_{min}}X_t^{\Phi 3'}\mathcal{G}(t)] \\ &= \left(1-\frac{\mathbb{E}[U_{min}]}{T}\right)\rho_w^{\zeta}(\mathbf{w}), \qquad (A.83)\end{aligned}$$

with $X_t^{\Phi 3'}$ given by (A.48). Substituting (A.82) and (A.83) into (A.81), we have

$$|\frac{1}{T}\mathbb{E}[\sum_{t=1}^{T}X_t^{\Phi 3}\mathcal{G}(t)]-\rho_w^{\zeta}(\mathbf{w})| \leq 2\left(1-\frac{\mathbb{E}[U_{min}]}{T}\right)\rho_w^{\zeta}(\mathbf{w}).$$

To see the convergence rate of $\mathbb{E}[U_{min}]/T$ to 1 as $T\to+\infty$, for any $\epsilon>0$ small,

$$1-\frac{\mathbb{E}[U_{min}]}{T} \leq \mathbb{P}(\frac{U_{min}}{T}<1-\epsilon)+\epsilon \leq (1+e)\exp(-T\Delta_U^2)+\epsilon, \qquad (A.84)$$

for $T>T_N$ from Lemma 4, with Δ_U given by (2.18). We want to compute the optimal achievable convergence rate given by (A.84). Denote $\Delta_U = D_\zeta\epsilon$, where D_ζ is a constant determined by ζ from (2.18). To compute the optimal upper bound of (A.84), set $\exp(-T\Delta_U^2)=\Delta_U$. The solution is characterized by the Lambert W function as

$$\Delta_U^* = \frac{1}{\exp(\tilde{W}_0(2T)/2)}, \qquad (A.85)$$

where $\tilde{W}_0(2T)$ is the upper branch of the Lambert W function $\tilde{W}_0\geq -1$. Next, we use the asymptotic approximation of $\tilde{W}_0(2T)$ for large T. [74] show that $\tilde{W}_0(2T)=\ln(2T)-\ln\ln(2T)+o(1)$ holds for large T, then (A.85) becomes

$$\Delta_U^* \approxeq \frac{1}{\exp\left(\frac{1}{2}\ln(2T)-\frac{1}{2}\ln\ln(2T)\right)} = O(\frac{\sqrt{\ln T}}{\sqrt{T}}).$$

Recall that $T_N = \lceil \frac{2/\epsilon+1}{1-\zeta} \rceil = O(1/\epsilon)$ and $T > T_N$ leads to $\epsilon > \frac{1}{T}$. Therefore, taking $\epsilon = \frac{1}{D_\zeta}\sqrt{\frac{\ln T}{T}}$, (A.84) becomes

$$1 - \frac{\mathbb{E}[U_{min}]}{T} \leq (1+e)\exp(-\ln T) + \frac{1}{D_\zeta}\frac{\sqrt{\ln T}}{\sqrt{T}} = O(\frac{\sqrt{\ln T}}{\sqrt{T}}).$$

Therefore, $R_w(\Phi 3)$ converges to $\rho_w^\zeta(\mathbf{w})$ with rate $O(\sqrt{\ln T}/\sqrt{T})$ as $T \to +\infty$.

For $r_j(\Phi 3)$, we use the same arguments as above with $\mathcal{G}(t)$ substituted by $\mathcal{A}_j(t)$. Specifically,

$$\begin{aligned}
|\frac{1}{T}\mathbb{E}[\sum_{t=1}^{T} X_t^{\Phi 3}\mathcal{A}_j(t)] - \rho_j^\zeta(w)| \leq & |\frac{1}{T}\mathbb{E}[\sum_{t=1}^{U_{min}} X_t^{\Phi 3}\mathcal{A}_j(t)] - \rho_j^\zeta(w)| \\
& + \frac{1}{T}\mathbb{E}[\mathbf{1}_{U_{min}<T}\sum_{t=U_{min}+1}^{T} X_t^{\Phi 3}\mathcal{A}_j(t)] \\
\leq & 2\left(1 - \frac{\mathbb{E}[U_{min}]}{T}\right)\rho_j^\zeta(w).
\end{aligned}$$

Therefore, the convergence rate of expected reward per task for $r_j(\Phi 3)$ follows from Lemma 4. ■

APPENDIX B

PROOFS FOR CHAPTER 5

B.1 Proof for Lemma 6

The proof is similar to the proof of Lemma 5. We prove the upper bound by providing upper bounds for two elements: (1) the number of job assignments each worker can complete, and (2) the expected job value that is assigned to each worker in a stage.

Since each job requires l_0 time slots to complete, an upper bound for the number of jobs each worker can complete is $(\lfloor T/l_0 \rfloor + 1)$. Note that jobs have IID values, so each time interval of length l_0 has the same distribution. Since only M workers are available, the optimal expected reward obtainable during any time interval of length l_0 has an upper bound given by the optimal reward for assigning the l_0 jobs to $\min\{M, l_0\}$ workers with the largest success rates. Therefore, from Theorem 25, an upper bound for the optimal expected reward during any time interval of length l_0 is $\sum_{i=l_0-\min\{l_0,M\}+1}^{l_0} a_i^{l_0} w_{M-l_0+i}$, which is achieved by assigning the job with the i^{th} largest value out of l_0 jobs to the worker with the i^{th} largest success rate (in expectation). Combining these two upper bounds leads to the desired result. ■

B.2 Proof for Proposition 5

The proof is similar to the proof of Proposition 4. Since job values are IID and the Greedy SSAP-stage algorithm assigns jobs arriving in each stage using the same rule, by Wald's identity, the expected reward using the Greedy SSAP-stage algorithm is the product of the expected number of stages and the expected reward in each stage. The lower bound for the total number of stages still holds (see (5.4)). Since the optimal policy for an l_0-depth SSAP problem

is used by the Greedy SSAP-stage algorithm in each stage, then from Theorem 25, the expected reward in each stage is $\sum_{i=l_0-\min\{l_0,M\}+1}^{l_0} a_i^{l_0} w_{M-l_0+i}$. Combining the lower bound for the expected number of stages and the expected reward in each stage completes the proof. ■

B.3 Proof for Lemma 8

First we show

$$\sum_{i=l_1-l+1}^{l_1} a_i^{l_1} - \sum_{i=l_1-l}^{l_1-1} a_i^{l_1-1} = \int_{a_{l_1-l}^{l_1-1}}^{+\infty} (x - a_{l_1-l}^{l_1-1}) \mathbb{P} dx. \tag{B.1}$$

We prove (B.1) by induction on l. If $l = 1$, then (B.1) is the same as (5.24) and hence holds from Lemma 7.

Suppose (B.1) holds for $l = k \geq 1$. Consider $l = k + 1$. Then from the recursive definitions of a_i^j in (5.1),

$$\begin{aligned}
& a_{l_1-k}^{l_1} - a_{l_1-k-1}^{l_1-1} \\
&= \int_{a_{l_1-k-1}^{l_1-1}}^{a_{l_1-k}^{l_1-1}} x \mathbb{P} dx + a_{l_1-k-1}^{l_1-1} \mathbb{P}(v < a_{l_1-k-1}^{l_1-1}) + a_{l_1-k}^{l_1-1} \mathbb{P}(v > a_{l_1-k}^{l_1-1}) - a_{l_1-k-1}^{l_1-1} \\
&= \int_{a_{l_1-k-1}^{l_1-1}}^{a_{l_1-k}^{l_1-1}} (x - a_{l_1-k-1}^{l_1-1}) \mathbb{P} dx + \int_{a_{l_1-k}^{l_1-1}}^{+\infty} (a_{l_1-k}^{l_1-1} - a_{l_1-k-1}^{l_1-1}) \mathbb{P} dx.
\end{aligned} \tag{B.2}$$

Therefore,

$$
\begin{aligned}
&\sum_{i=l_1-k}^{l_1} a_i^{l_1} - \sum_{i=l_1-k-1}^{l_1-1} a_i^{l_1-1} \\
=&a_{l_1-k}^{l_1} - a_{l_1-k-1}^{l_1-1} + \sum_{i=l_1-k+1}^{l_1} a_i^{l_1} - \sum_{i=l_1-k}^{l_1-1} a_i^{l_1-1} \\
\overset{(B.2)}{=}& \int_{a_{l_1-k-1}^{l_1-1}}^{a_{l_1-k}^{l_1-1}} (x - a_{l_1-k-1}^{l_1-1})\mathbb{P}dx + \int_{a_{l_1-k}^{l_1-1}}^{+\infty} (a_{l_1-k}^{l_1-1} - a_{l_1-k-1}^{l_1-1})\mathbb{P}dx \\
&+ \sum_{i=l_1-k+1}^{l_1} a_i^{l_1} - \sum_{i=l_1-k}^{l_1-1} a_i^{l_1-1} \\
=& \int_{a_{l_1-k-1}^{l_1-1}}^{a_{l_1-k}^{l_1-1}} (x - a_{l_1-k-1}^{l_1-1})\mathbb{P}dx + \int_{a_{l_1-k}^{l_1-1}}^{+\infty} (a_{l_1-k}^{l_1-1} - a_{l_1-k-1}^{l_1-1})\mathbb{P}dx + \int_{a_{l_1-k}^{l_1-1}}^{+\infty} (x - a_{l_1-k}^{l_1-1})\mathbb{P}dx \\
=& \int_{a_{l_1-k-1}^{l_1-1}}^{+\infty} (x - a_{l_1-k-1}^{l_1-1})\mathbb{P}dx,
\end{aligned}
$$

where the second to last equality follows from induction assumption, and hence (B.1) holds.

Substituting l_1 with $(l_2 + 1)$ in (B.1) leads to

$$\sum_{i=l_2-l+2}^{l_2+1} a_i^{l_2+1} - \sum_{i=l_2-l+1}^{l_2} a_i^{l_2} = \int_{a_{l_2-l+1}^{l_2}}^{+\infty} (x - a_{l_2-l+1}^{l_2})\mathbb{P}dx. \tag{B.3}$$

Since $l_2 - (l_2 - l + 1) = (l_1 - 1) - (l_1 - l) = l - 1$ and $l_2 > l_1 - 1$, then $a_{l_2-l+1}^{l_2} \geq a_{l_1-l}^{l_1-1}$ (since $a_{l_2-l+1}^{l_2}(a_{l_1-l}^{l_1-1})$ is the expected value of the l^{th} largest order statistics of $l_2(l_1 - 1)$ IID samples). Combining (B.1) and (B.3) leads to

$$\sum_{i=l_1-l+1}^{l_1} a_i^{l_1} - \sum_{i=l_1-l}^{l_1-1} a_i^{l_1-1} \geq \sum_{i=l_2-l+2}^{l_2+1} a_i^{l_2+1} - \sum_{i=l_2-l+1}^{l_2} a_i^{l_2},$$

which completes the proof. ■

B.4 Proof for Corollary 5

We first prove $h_1 \sum_{i=\lfloor T/h_1 \rfloor - l + 1}^{\lfloor T/h_1 \rfloor} a_i^{\lfloor T/h_1 \rfloor} \geq (h_1 - 1) \sum_{i=\lfloor T/(h_1-1) \rfloor - l + 1}^{\lfloor T/(h_1-1) \rfloor} a_i^{\lfloor T/(h_1-1) \rfloor}$ for any $h_1 > 1$ and $l \leq T/(2h_1)$. Construct h_1 groups of IID job values

(samples) as follows: (h_1-2) groups with $\lfloor T/(h_1-1)\rfloor$ job values, one group with $(\lfloor T/(h_1-1)\rfloor - l)$ job values, and one group with l job values. Therefore,

$$\begin{aligned}
&h_1 \sum_{i=\lfloor T/h_1\rfloor - l+1}^{\lfloor T/h_1\rfloor} a_i^{\lfloor T/h_1\rfloor} \\
\geq &(h_1-2) \sum_{i=\lfloor T/(h_1-1)\rfloor - l+1}^{\lfloor T/(h_1-1)\rfloor} a_i^{\lfloor T/(h_1-1)\rfloor} + \sum_{i=\lfloor T/(h_1-1)\rfloor - 2l+1}^{\lfloor T/(h_1-1)\rfloor - l} a_i^{\lfloor T/(h_1-1)\rfloor - l} + \sum_{i=1}^{l} a_i^l \qquad \text{(B.4)} \\
\geq &(h_1-2) \sum_{i=\lfloor T/(h_1-1)\rfloor - l+1}^{\lfloor T/(h_1-1)\rfloor} a_i^{\lfloor T/(h_1-1)\rfloor} + \sum_{i=\lfloor T/(h_1-1)\rfloor - l+1}^{\lfloor T/(h_1-1)\rfloor} a_i^{\lfloor T/(h_1-1)\rfloor} \\
\geq &(h_1-1) \sum_{i=\lfloor T/(h_1-1)\rfloor - l+1}^{\lfloor T/(h_1-1)\rfloor} a_i^{\lfloor T/(h_1-1)\rfloor},
\end{aligned}$$

where (a) the first inequality is an application of Lemma 8: when the numbers of samples and divided groups are fixed, the sum of the expected value of the l largest order statistics in each group is maximized by dividing the samples into equal-size groups. We assume $l \leq T/(2h_1)$ and hence $\lfloor T/(h_1-1)\rfloor \geq 2l$, which only increases the right-hand side of the (B.4) without influencing the direction of the inequality of (B.4). (b) The second inequality follows from the definition of order statistics, and hence the sum of the expected value of the l largest order statistics of $\lfloor T/(h_1-1)\rfloor$ IID job value samples is no greater than the sum of the expected value of the l largest order statistics of $\lfloor T/(h_1-1)\rfloor - l$ IID job value samples and the l largest order statistics of l IID job value samples. That is,

$$\sum_{i=\lfloor T/(h_1-1)\rfloor - l+1}^{\lfloor T/(h_1-1)\rfloor} a_i^{\lfloor T/(h_1-1)\rfloor} \leq \sum_{i=\lfloor T/(h_1-1)\rfloor - 2l+1}^{\lfloor T/(h_1-1)\rfloor - l} a_i^{\lfloor T/(h_1-1)\rfloor - l} + \sum_{i=1}^{l} a_i^l.$$

Since $h_1, h_2 \in \mathbb{Z}^+$, $h_1 > h_2$ and $l \leq T/(2h_1)$, then

$$\begin{aligned} h_1 \sum_{i=\lfloor T/h_1 \rfloor - l + 1}^{\lfloor T/h_1 \rfloor} a_i^{\lfloor T/h_1 \rfloor} &\geq (h_1 - 1) \sum_{i=\lfloor T/(h_1-1) \rfloor - l + 1}^{\lfloor T/(h_1-1) \rfloor} a_i^{\lfloor T/(h_1-1) \rfloor} \\ &\geq \dots \\ &\geq h_2 \sum_{i=\lfloor T/h_2 \rfloor - l + 1}^{\lfloor T/h_2 \rfloor} a_i^{\lfloor T/h_2 \rfloor}. \end{aligned}$$

■

B.5 Proof for Proposition 7

Since jobs have IID values and memoryless lengths, we compute the expected reward using the Greedy SSAP-stage algorithm by considering the expected total number of stages and the expected reward in each stage.

Let N_L denote the number of stages. Let L_i denote the length of stage i, $i = 1, 2, \dots, N_L$. Then the $\{L_i\}$ are IID random variables with expectation $\mathbb{E}[L_i] = \lceil 1/q \rceil + \lceil 1/q \rceil = 2\lceil 1/q \rceil$, which is the sum of a constant (length of the first-half stage when workers wait to be assigned jobs) and the expectation of a geometrically distributed random variable (length of the second-half stage when workers complete assigned jobs). Note that the number of stages by any time $t > 0$ is a renewal process, and hence by definition N_L is a stopping rule for $\{L_i\}$. Using Wald's identity leads to

$$\sum_{i=1}^{N_L} L_i \geq T \qquad \Rightarrow \mathbb{E}[\sum_{i=1}^{N_L} L_i] \geq T \qquad \Rightarrow \mathbb{E}[N_L]\mathbb{E}[L_i] \geq T,$$

and hence $\mathbb{E}[N_L] \geq T/(2\lceil 1/q \rceil)$.

Let R_i denote the reward for stage i, $i = 1, 2, \dots, N_L$. Then the $\{R_i\}$ are IID random variables. Since the optimal policy for a $\lceil 1/q \rceil$-depth SSAP problem is used, then from Theorem 25,

$$\mathbb{E}[R_i] = \sum_{i=\lceil 1/q \rceil - \min\{\lceil 1/q \rceil, M\} + 1}^{\lceil 1/q \rceil} a_i^{\lceil 1/q \rceil} w_{M - \lceil 1/q \rceil + i}.$$

Then using Wald's identity,

$$\mathbb{E}[R_{MLL}] = \mathbb{E}[\sum_{i=1}^{N_L} R_i] = \mathbb{E}[N_L]\mathbb{E}[R_i]$$

$$\geq T/(2\lceil 1/q \rceil) \sum_{i=\lceil 1/q \rceil - \min\{\lceil 1/q \rceil, M\}+1}^{\lceil 1/q \rceil} a_i^{\lceil 1/q \rceil} w_{M-\lceil 1/q \rceil + i}.$$

■

B.6 Proof for Theorem 33

Similarly to the proof of Theorem 32, we consider the conditional expected reward for an assignment given by the optimal weighted matching at time $t \geq \lceil 1/q \rceil$ and the probability of such an assignment being feasible for memoryless-length jobs, on the condition that a real job arrives at time t.

Let OPT denote the optimal offline reward for scheduling T real jobs. Suppose that a real job J_t arrives at time t. Let $W_t = [t - \lceil 1/q \rceil + 1, t]$ denote the rolling window used by the rolling window algorithm for assigning J_t. Let N_W denote the number of real job arrivals during $\hat{W}_t = [t - \lceil 1/q \rceil + 1, t-1]$. Then N_W is a binomial random variable with parameter $(\lceil 1/q \rceil - 1)$ and p, with $\mathbb{P}(N_W = x) = \binom{\lceil 1/q \rceil - 1}{x} p^x (1-p)^{\lceil 1/q \rceil - x - 1}$, $x = 0, 1, \ldots, \lceil 1/q \rceil - 1$. (5.43) still holds for the conditional expected reward for the assignment at time t given a real job arrives at time t (substituting the length of rolling windows with $\lceil 1/q \rceil$).

Suppose the optimal weighted matching on workers and jobs arriving during rolling window $W_t = [t - \lceil 1/q \rceil + 1, t]$ assigns job J_t to worker w_{m_t}. We compute the conditional probability of this assignment at time t being feasible given that a real job J_t arrives at time t, that is, the conditional probability that worker w_{m_t} is available at time t, given a real job arrives at time t.

Note that the upper bound (5.34) for the probability that worker w_{m_t} is assigned at some time $t' \in [t - \lceil 1/q \rceil + 1, t-1]$ still holds. Then, substituting l_0 for $\lceil 1/q \rceil$ and following the same argument (5.45) to (5.49) lead to the upper bound for the conditional probability that worker w_{m_t} is available given that

w_{m_t} is not assigned before W_t and a job arrives at time t

$$\mathbb{P}(\text{worker } w_{m_t} \text{ is aviable at } t|\text{a job arrives at } t) \geq e^{-p}, \tag{B.5}$$

for $\lceil 1/q \rceil \geq 2p$. Since the job length follows a geometric distribution with parameter q, if worker w_{m_t} is not assigned in $[t - \lceil 1/q \rceil + 1, t]$ but assigned before rolling window W_t, there is still a probability that worker w_{m_t} has not completed the assigned job. The lower bound (5.35) for the conditional probability that worker w_{m_t} is available before time t given that w_{m_t} is assigned before W_t still holds. Let $Event_t$ and $Event_{W_t}$ denote the event of a job arriving at time t and the worker w_{m_t} assigned before rolling window W_t, respectively. Then,

$$\begin{aligned}
&\mathbb{P}(\text{worker } w_{m_t} \text{ is available at } t|Event_t) \\
\geq&\mathbb{P}(\text{worker } w_{m_t} \text{ is not assigned in } [t - \lceil 1/q \rceil + 1, t-1]|Event_t) \\
&\times \Big(\mathbb{P}(\text{not } Event_{W_t}|Event_t) \\
&+ \mathbb{P}(\text{worker } w_{m_t} \text{ is available before } t|Event_{W_t}, Event_t)\mathbb{P}(Event_{W_t}|Event_t)\Big) \\
\geq&\mathbb{P}(\text{worker } w_{m_t} \text{is not assigned in } [t - \lceil 1/q \rceil + 1, t-1]|Event_t) \\
&\times \mathbb{P}(\text{worker } w_{m_t} \text{ is available before time } t|Event_{W_t}, Event_t) \\
=&\mathbb{P}(\text{worker } w_{m_t} \text{is not assigned in } [t - \lceil 1/q \rceil + 1, t-1]|Event_t) \\
&\times \mathbb{P}(\text{worker } w_{m_t} \text{ is available before time } t|Event_{W_t}) \qquad \text{(B.6)} \\
\overset{\text{(B.5),(5.35)}}{\geq}& e^{-p}(1 - (1-q)^{\lceil 1/q \rceil}) \\
\geq&\frac{e-1}{e^{p+1}}, \qquad \text{(B.7)}
\end{aligned}$$

where equality (B.6) follows since $Event_t$ and $Event_{W_t}$ are independent.

Let R_{RW}^{Mp} denote the reward for the rolling window algorithm for memoryless-length jobs with arrivals that are geometric with parameter p and randomly ordered. Let T_p denote the time slot when the last real job arrives. Since the inter-arrival time between subsequent real jobs is a geometric random variable with parameter p, T_p is a random variable with $\mathbb{E}[T_p] = T\lceil 1/p \rceil$.

Combining (5.43) and (B.7) leads to

$$
\begin{aligned}
&\mathbb{E}[R_{RW}^{Mp}] \\
=&\mathbb{E}\Bigg[\sum_{t=1}^{T_p}\mathbb{E}[R(t)|Event_t]\times\mathbb{P}(\text{worker } w_{m_t}\text{is available at } t|Event_t) \\
&\times\mathbb{P}(Event_t)\Bigg] \\
\geq&\mathbb{E}\Bigg[\sum_{t=\lceil 1/q\rceil}^{T_p}\mathbb{E}[R(t)|Event_t]\times\mathbb{P}(\text{worker } w_{m_t}\text{is available at } t|Event_t) \\
&\times\mathbb{P}(Event_t)\Bigg] \\
\overset{(5.43),(B.7)}{\geq}&\mathbb{E}\left[\sum_{t=\lceil 1/q\rceil}^{T_p}\frac{OPT}{T}\frac{e-1}{e^{p+1}}p\right] \\
=&\frac{OPT}{T}\frac{e-1}{e^{p+1}}p(\mathbb{E}[T_p]-\lceil 1/q\rceil+1) \qquad \text{(B.8)} \\
\to&\frac{(e-1)OPT}{e^{p+1}},
\end{aligned}
$$

as $T\to+\infty$, which completes the proof. ■

B.7 BOM Algorithm

Algorithm 7 BOM Algorithm (Kesselheim et al. 2013)

Let L' denote the first $\lfloor T/e\rfloor$ vertices of L.
Let $M=\emptyset$.
for each subsequent vertex $l\in L\setminus L'$ **do**
  $L'=l\bigcup L'$.
  $M^{(l)}=$ optimal matching on $G(L'\bigcup R)$.
  Let $e^{(l)}$ denote the edge assigned to l in $M^{(l)}$.
  if $M\bigcup e^{(l)}$ is a valid matching **then** match l as $M=M\bigcup e^{(l)}$.
  end if
end for

REFERENCES

[1] I. David, "A sequential assignment match process with general renewal arrival times," *Probability in the Engineering and Informational Sciences*, vol. 9, no. 03, pp. 475–492, 1995.

[2] X. Su and S. A. Zenios, "Patient choice in kidney allocation: A sequential stochastic assignment model," *Operations Research*, vol. 53, no. 3, pp. 443–455, 2005.

[3] R. Kleinberg, "A multiple-choice secretary algorithm with applications to online auctions," in *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 2005, pp. 630–631.

[4] L. McLay, S. Jacobson, and A. Nikolaev, "A sequential stochastic passenger screening problem for aviation security," *IIE Transactions*, vol. 41, no. 6, pp. 575–591, 2009.

[5] L. McLay, A. Lee, and S. Jacobson, "Risk-based policies for airport security checkpoint screening," *Transportation Science*, vol. 44, no. 3, pp. 333–349, 2010.

[6] "Fact sheet: Screening and monitoring travelers to prevent the spread of Ebola," http://www.cdc.gov/vhf/ebola/travelers/ebola-screening-factsheet.html, Jun 2015d, retrieved Aug. 27, 2015.

[7] S. H. Jacobson, G. Yu, and J. A. Jokela, "A double-risk monitoring and movement restriction policy for Ebola entry screening at airports in the United States," *Preventive Medicine*, vol. 88, pp. 33–38, 2016.

[8] C. Derman, G. Lieberman, and S. Ross, "A sequential stochastic assignment problem," *Management Science*, vol. 18, no. 7, pp. 349–355, 1972.

[9] S. Albright, "Optimal sequential assignments with random arrival times," *Management Science*, vol. 21, no. 1, pp. 60–67, 1974.

[10] A. Nikolaev and S. Jacobson, "Technical note — Stochastic sequential decision-making with a random number of jobs," *Operations Research*, vol. 58, no. 4-part-1, pp. 1023–1027, 2010.

[11] D. Kennedy, "Optimal sequential assignment," *Mathematics of Operations Research*, vol. 11, no. 4, pp. 619–626, 1986.

[12] S. Albright, "A Markov chain version of the secretary problem," *Naval Research Logistics Quarterly*, vol. 23, no. 1, pp. 151–159, 1976.

[13] T. Nakai, "A sequential stochastic assignment problem in a partially observable markov chain," *Mathematics of Operations Research*, vol. 11, no. 2, pp. 230–240, 1986.

[14] F. T. Bruss and T. S. Ferguson, "Multiple buying or selling with vector offers," *Journal of Applied Probability*, vol. 34, no. 4, pp. 959–973, 1997.

[15] S. M. Ross and D. T. Wu, "A generalized coupon collecting model as a parsimonious optimal stochastic assignment model," *Annals of Operations Research*, vol. 208, no. 1, pp. 133–146, 2013.

[16] D. T. Wu and S. M. Ross, "A stochastic assignment problem," *Naval Research Logistics (NRL)*, vol. 62, no. 1, pp. 23–31, 2015.

[17] A. Lee, L. McLay, and S. Jacobson, "Designing aviation security passenger screening systems using nonlinear control," *SIAM Journal on Control and Optimization*, vol. 48, no. 4, pp. 2085–2105, 2009.

[18] A. Nikolaev, A. Lee, and S. Jacobson, "Optimal aviation security screening strategies with dynamic passenger risk updates," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, no. 1, pp. 203–212, 2012.

[19] S. A. Zenios, G. M. Chertow, and L. M. Wein, "Dynamic allocation of kidneys to candidates on the transplant waiting list," *Operations Research*, vol. 48, no. 4, pp. 549–569, 2000.

[20] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani, "Adwords and generalized online matching," *Journal of the ACM (JACM)*, vol. 54, no. 5, p. 22, 2007.

[21] N. R. Devanur, K. Jain, B. Sivan, and C. A. Wilkens, "Near optimal online algorithms and fast approximation algorithms for resource allocation problems," in *Proceedings of the 12th ACM conference on Electronic commerce*. ACM, 2011, pp. 29–38.

[22] T. Kesselheim, A. Tönnis, K. Radke, and B. Vöcking, "Primal beats dual on online packing lps in the random-order model," in *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*. ACM, 2014, pp. 303–312.

[23] J. Feldman, A. Mehta, V. Mirrokni, and S. Muthukrishnan, "Online stochastic matching: Beating 1-1/e," in *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on.* IEEE, 2009, pp. 117–126.

[24] M. Mahdian and Q. Yan, "Online bipartite matching with random arrivals: an approach based on strongly factor-revealing lps," in *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing.* ACM, 2011, pp. 597–606.

[25] C. Albright and C. Derman, "Asymptotic optimal policies for the stochastic sequential assignment problem," *Management Science*, vol. 19, no. 1, pp. 46–51, 1972.

[26] G. Baharian and S. Jacobson, "Limiting behavior of the stochastic sequential assignment problem," *Naval Research Logistics (NRL)*, vol. 60, no. 4, pp. 321–330, 2013.

[27] G. Baharian and S. Jacobson, "Limiting behavior of the target-dependent stochastic sequential assignment problem," *Journal of Applied Probability*, vol. 51, no. 4, pp. 943–953, 2014.

[28] F. B. Abdelaziz, "Solution approaches for the multiobjective stochastic programming," *European Journal of Operational Research*, vol. 216, no. 1, pp. 1–16, 2012.

[29] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, 2013.

[30] M. Ehrgott, *Multicriteria Optimization.* Berlin, Heidelberg, Germany: Springer Science & Business Media, 2006.

[31] J.-F. Bérubé, M. Gendreau, and J. Potvin, "An exact epsilon-constraint method for bi-objective combinatorial optimization problems: Application to the traveling salesman problem with profits," *European Journal of Operational Research*, vol. 194, no. 1, pp. 39–50, 2009.

[32] E. M. Arkin and E. B. Silverberg, "Scheduling jobs with fixed start and end times," *Discrete Applied Mathematics*, vol. 18, no. 1, pp. 1–8, 1987.

[33] E. L. Lawler, "A dynamic programming algorithm for preemptive scheduling of a single machine to minimize the number of late jobs," *Annals of Operations Research*, vol. 26, no. 1, pp. 125–133, 1990.

[34] A. Bar-Noy, S. Guha, J. Naor, and B. Schieber, "Approximating the throughput of multiple machines in real-time scheduling," *SIAM Journal on Computing*, vol. 31, no. 2, pp. 331–352, 2001.

[35] T. Erlebach and F. C. Spieksma, "Simple algorithms for a weighted interval selection problem," in *International Symposium on Algorithms and Computation*. Springer, 2000, pp. 228–240.

[36] L. Epstein, J. Sgall et al., "Approximation schemes for scheduling on uniformly related and identical parallel machines," *Algorithmica*, vol. 39, no. 1, pp. 43–57, 2004.

[37] S. O. Krumke, C. Thielen, and S. Westphal, "Interval scheduling on related machines," *Computers & Operations Research*, vol. 38, no. 12, pp. 1836–1844, 2011.

[38] U. Faigle and W. M. Nawijn, "Note on scheduling intervals on-line," *Discrete Applied Mathematics*, vol. 58, no. 1, pp. 13–17, 1995.

[39] G. J. Woeginger, "On-line scheduling of jobs with fixed start and end times," *Theoretical Computer Science*, vol. 130, no. 1, pp. 5–16, 1994.

[40] S. P. Fung, C. K. Poon, and F. Zheng, "Online interval scheduling: randomized and multiprocessor cases," *Journal of Combinatorial Optimization*, vol. 16, no. 3, pp. 248–262, 2008.

[41] S. Baruah, G. Koren, D. Mao, B. Mishra, A. Raghunathan, L. Rosier, D. Shasha, and F. Wang, "On the competitiveness of on-line real-time task scheduling," *Real-Time Systems*, vol. 4, no. 2, pp. 125–144, 1992.

[42] H. Miyazawa and T. Erlebach, "An improved randomized on-line algorithm for a weighted interval selection problem," *Journal of Scheduling*, vol. 7, no. 4, pp. 293–311, 2004.

[43] M. Y. Kovalyov, C. Ng, and T. E. Cheng, "Fixed interval scheduling: Models, applications, computational complexity and algorithms," *European Journal of Operational Research*, vol. 178, no. 2, pp. 331–342, 2007.

[44] S. S. Seiden, "Randomized online interval scheduling," *Operations Research Letters*, vol. 22, no. 4, pp. 171–177, 1998.

[45] S. P. Fung, C. K. Poon, and F. Zheng, "Improved randomized online scheduling of intervals and jobs," *Theory of Computing Systems*, vol. 55, no. 1, pp. 202–228, 2014.

[46] L. Epstein, Ł. Jeż, J. Sgall, and R. van Stee, "Online scheduling of jobs with fixed start times on related machines," *Algorithmica*, vol. 74, no. 1, pp. 156–176, 2016.

[47] S. P. Fung, C. K. Poon, and D. K. Yung, "On-line scheduling of equal-length intervals on parallel machines," *Information Processing Letters*, vol. 112, no. 10, pp. 376–379, 2012.

[48] U. Faigle, W. Kern, and W. M. Nawijn, "A greedy on-line algorithm for thek-track assignment problem," *Journal of Algorithms*, vol. 31, no. 1, pp. 196–210, 1999.

[49] B. DasGupta and M. A. Palis, "Online real-time preemptive scheduling of jobs with deadlines on multiple machines," *Journal of Scheduling*, vol. 4, no. 6, pp. 297–312, 2001.

[50] D. Gross, *Fundamentals of Queueing Theory*. Hoboken, New Jersey: John Wiley & Sons, 2008.

[51] B. Bahmani and M. Kapralov, "Improved bounds for online stochastic matching," *Algorithms–ESA 2010*, pp. 170–181, 2010.

[52] C. Karande, A. Mehta, and P. Tripathi, "Online bipartite matching with unknown distributions," in *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*. ACM, 2011, pp. 587–596.

[53] T. Kesselheim, K. Radke, A. Tönnis, and B. Vöcking, "An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions," in *European Symposium on Algorithms*. Springer, 2013, pp. 589–600.

[54] A. Fiat, I. Gorelik, H. Kaplan, and S. Novgorodov, "The temp secretary problem," in *Algorithms-ESA 2015*. Springer, 2015, pp. 631–642.

[55] N. Buchbinder, J. S. Naor et al., "The design of competitive online algorithms via a primal–dual approach," *Foundations and Trends in Theoretical Computer Science*, vol. 3, no. 2–3, pp. 93–263, 2009.

[56] M. Wang and Y. Chen, "An online primal-dual method for discounted markov decision processes," in *Decision and Control (CDC), 2016 IEEE 55th Conference on*. IEEE, 2016, pp. 4516–4521.

[57] N. Bansal, N. Buchbinder, and J. S. Naor, "A primal-dual randomized algorithm for weighted paging," *Journal of the ACM (JACM)*, vol. 59, no. 4, p. 19, 2012.

[58] S. Agrawal, Z. Wang, and Y. Ye, "A dynamic near-optimal algorithm for online linear programming," *Operations Research*, vol. 62, no. 4, pp. 876–890, 2014.

[59] N. Buchbinder, K. Jain, and M. Singh, "Secretary problems via linear programming," in *International Conference on Integer Programming and Combinatorial Optimization*. Springer, 2010, pp. 163–176.

[60] N. R. Devanur, K. Jain, and R. D. Kleinberg, "Randomized primal-dual analysis of ranking for online bipartite matching," in *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2013, pp. 101–107.

[61] R. M. Karp, U. V. Vazirani, and V. V. Vazirani, "An optimal algorithm for on-line bipartite matching," in *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*. ACM, 1990, pp. 352–358.

[62] Y. Chen and M. Wang, "Stochastic primal-dual methods and sample complexity of reinforcement learning," *arXiv:1612.02516*, 2016.

[63] F. C. Spieksma, "On the approximability of an interval scheduling problem," *Journal of Scheduling*, vol. 2, no. 5, pp. 215–227, 1999.

[64] K. Miettinen, *Nonlinear Multiobjective Optimization*. New York: Springer Science & Business Media, 1999.

[65] G. Yu, S. H. Jacobson, and N. Kiyavash, *A bi-criteria multiple-choice secretary problem*. Technical Report, University of Illinois at Urbana-Champaign, 2016.

[66] A. S. Schulz and M. Skutella, "Scheduling unrelated machines by randomized rounding," *SIAM Journal on Discrete Mathematics*, vol. 15, no. 4, pp. 450–469, 2002.

[67] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific Belmont, MA, 1995.

[68] G. Hardy, J. Littlewood, and G. Pólya, *Inequalities*. Cambridge University Press, 1952.

[69] A. DasGupta, "Finite sample theory of order statistics and extremes," in *Probability for Statistics and Machine Learning*. Springer, 2011, pp. 221–248.

[70] G. Yu and S. H. Jacobson, "Online c-benevolent job scheduling on multiple machines," *Optimization Letters*, vol. 12, no. 2, pp. 251–263, 2018.

[71] D. Blackwell, "On an equation of Wald," *The Annals of Mathematical Statistics*, pp. 84–87, 1946.

[72] H. Anton, I. Bivens, S. Davis, and T. Polaski, *Calculus*. Wiley, 2002.

[73] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 13–30, 1963.

[74] R. Corless, G. Gonnet, D. Hare, D. Jeffrey, and D. Knuth, "On the LambertW function," *Advances in Computational Mathematics*, vol. 5, no. 1, pp. 329–359, 1996.