Hindawi
Journal of Mathematics
Volume 2023, Article ID 8886388, 8 pages
https://doi.org/10.1155/2023/8886388

Research Article

A Combinatorial Approximation Algorithm for the Vector Scheduling with Submodular Penalties on Parallel Machines

Bihui Cheng[1] **and Wencheng Wang**[2]

[1]*School of Preparatory Education, Yunnan Minzu University, Kunming 650504, China*
[2]*School of Mathematics and Computer Science, Yunnan Minzu University, Kunming 650504, China*

Correspondence should be addressed to Wencheng Wang; wencheng@ymu.edu.cn

Received 15 December 2022; Revised 19 February 2023; Accepted 14 November 2023; Published 29 November 2023

Academic Editor: Ji Gao

In this paper, we focus on solving the vector scheduling problem with submodular penalties on parallel machines. We are given n jobs and m parallel machines, where each job is associated with a d-dimensional vector. Each job can either be rejected, incurring a rejection penalty, or accepted and processed on one of the m parallel machines. The objective is to minimize the sum of the maximum load overall dimensions of the total vector for all accepted jobs, along with the total penalty for rejected jobs. The penalty is determined by a submodular function. Our main work is to design a $(2-1/m)(\min\{r,d\})$-approximation algorithm to solve this problem. Here, r denotes the maximum ratio of the maximum load to the minimum load on the d-dimensional vectors among all jobs.

1. Introduction

The multiprocessor scheduling problem was first studied in 1966 by Graham [1]. It is an important problem in the field of combinatorial optimization [2, 3]. In this problem, we have a set of n jobs, denoted as $J=\{J_1,J_2,\ldots,J_n\}$, and a set of m parallel machines, denoted as $M=\{M_1,M_2,\ldots,M_m\}$. Each job can be processed on one of the machines. The objective is to minimize the makespan, which is the maximum completion time among all machines. The problem is strongly NP-hard [1, 4], meaning that it is computationally challenging to find an optimal solution.

In 1966, Graham [1] proposed a classical (2-1/m)-approximation algorithm called LS (list scheduling) based on a greedy strategy. Later, Graham [5] designed a 4/3-approximation algorithm called LPT (longest processing times) by sorting the jobs in nonincreasing order. In 2020, Jansen et al. [6] introduced an efficient polynomial-time approximation scheme, which is currently the best-known result for this problem.

In some scenarios, rejecting certain low-cost performance jobs can increase profits. Based on the multiprocessor scheduling problem, Bartal et al. [7] explored the problem of multiprocessor scheduling with rejection in 2000. This variant allows each job $J_j \in J$ to either be rejected with a penalty or scheduled on one of the machines. The objective is to minimize the makespan of the accepted jobs while also considering the total penalty incurred by the rejected jobs. They proposed a $(2-1/m)$-approximation algorithm. Later, Ou et al. [8] developed an improved $(3/2+\varepsilon)$-approximation algorithm, where $\varepsilon>0$ is a given small constant.

The vector scheduling problem, introduced by Chekuri and Khanna [9], is a further extension of the multiprocessor scheduling problem. In this problem, each job is associated with a d-dimensional vector. The objective is to schedule n d-dimensional jobs on m machines to minimize the maximum load overall dimensions of the machines. Chekuri and Khanna [9] proved in 2004 that this problem cannot be approximated within a constant factor unless NP = ZPP, assuming arbitrary input dimension d. Here, ZPP refers to a complexity class that deals with probabilistic Turing machines [10], which take into account the probability of acceptance. In the same paper, Chekuri and Khanna [9] also presented an $O(\ln^2 d)$-approximation algorithm and an $O(\ln \mathrm{dm}/\ln\ln \mathrm{dm})$-approximation algorithm with high probability, where m denotes the number of machines.

Recently, Harris and Srinivasan [11] developed a randomized polynomial-time approximation algorithm for the vector scheduling problem, achieving a ratio of $O(\ln\ d/\ln\ \ln\ d)$. For the vector scheduling problem with rejection, when $m = 1$, Dai and Li [12] demonstrated that the problem is NP-hard and proposed a d-approximation algorithm for a fixed constant d. Li and Cui [13] introduced a 2.54-approximation algorithm based on randomized rounding in the case where $m = 2$. Additional related results can be found in reference [14].

In many practical scenarios, the relationship between the number of rejected objects and the associated rejection penalties is often submodular rather than linear. As a result, combinatorial optimization problems with submodular penalties have gained significant attention. These problems involve a rejection penalty determining a submodular function [15, 16].

Liu and Li [17] addressed the multiprocessor scheduling problem with submodular penalties. They presented a combinatorial $(2 - 1/m)$-approximation algorithm based on the list scheduling (LS) algorithm and a greedy method. Wang and Liu [18] focused on parallel-machine scheduling with release times and submodular penalties, proposing a combinatorial 2-approximation algorithm. Zhang et al. [19] tackled the precedence-constrained scheduling problem with submodular rejection on parallel machines. They introduced a combinatorial 3-approximation algorithm to solve this problem. Liu et al. [20] considered the single-machine vector scheduling problem with submodular penalties. They presented a combinatorial $\min\{r, d\}$-approximation algorithm to solve this problem, where r represents the maximum ratio of the maximum load to the minimum load on the d-dimensional vectors among all jobs.

We are inspired by previous research studies and focus on the problem of vector scheduling with submodular penalties on parallel machines. We propose a $(2 - 1/m)(\min\{r, d\})$-approximation algorithm, where r represents the maximum ratio between the largest and smallest components of the d-dimensional vectors among all jobs. This algorithm extends the findings of prior results [17, 20].

This paper is organized as follows: In Section 2, we provide a formal problem statement and a fundamental lemma to ensure the correctness of our approximation algorithm. In Section 3, we present an approximation algorithm for the problem. In Section 4, we present our conclusions.

2. Preliminaries

Definition 1 (see [15]). Let U be a given set and π: $2^U \longrightarrow \mathbb{R}_{\geq 0}$ be a real-valued function defined on all subsets of U. It is called a submodular function if

$$\pi(A \cup B) + \pi(A \cap B) \leq \pi(A) + \pi(B), \forall A, B \subseteq U. \tag{1}$$

The problem of vector scheduling with submodular penalties on parallel machines (the VSSP-PM problem, for short) is defined as follows.

Given a set $J = \{J_1, J_2, \ldots, J_n\}$ of n jobs and a set $M = \{M_1, M_2, \ldots, M_m\}$ of m parallel machines, each job $J_j \in J$ is associated with a d-dimensional vector $\mathbf{p}_j = (p_j^1, p_j^2, \ldots, p_j^d)$, where $p_j^k \geq 0$ for all dimensions $k \in [d]$: $=\{1, 2, \cdots, d\}$. It is asked to find a scheduling configuration $(A_1, A_2, \ldots, A_m, R)$ that satisfies the following conditions: $A_1, A_2, \ldots, A_m$ form a partition of the accepted job set A (i.e., J$\backslash R$), where each A_i represents the set of jobs processed on machine M_i and R is the set of rejected jobs. The accepted job sets $A_1, A_2, \ldots, A_m$ are mutually exclusive, and their union covers the entire accepted job set A. The objective is to minimize the maximum load overall dimensions and the penalty incurred by the rejected jobs. The penalty, denoted as $\pi(R)$, is determined by a submodular function $\pi(\cdot)$. That is,

$$\min\left\{\max_i \max_k \sum_{j \in A_i} p_j^k + \pi(R)\right\}. \tag{2}$$

To further understand our problem, we present the following integer linear program (ILP) for the VSSP-PM problem.

$$\begin{aligned} &\min L + \sum_{R:R\subseteq J} \pi(R) z_R, \\ &\text{s.t.} \begin{cases} \sum_{i=1}^{m} x_j^i + \sum_{j:J_j \in R} z_R = 1, & \forall J_j \in J, \\ \sum_{j=1}^{n} p_j^k x_j^i \leq L, & \forall i \in [m], k \in [d], \\ x_j^i, z_R \in \{0, 1\}, & \forall R \subseteq J, i \in [m], j \in [n], \end{cases} \end{aligned} \tag{3}$$

where variable $z_R \in \{0, 1\}$ indicates whether R is picked, that is, $z_R = 1$ if and only if R is rejected, variable $x_j^i \in \{0, 1\}$ implies whether job $J_j \in J$ is accepted and scheduled on the machine M_i, here $x_j^i = 1$ if and only if J_j is accepted and scheduled on the machine M_i. Clearly, if $d = 1$, the VSSP-PM problem is exactly the problem of parallel machine scheduling with submodular penalties studied in Liu and Li [17]. If $m = 1$, the VSSP-PM problem becomes the problem of single machine vector scheduling with general penalties studied in Liu et al. [20].

For convenience, we may assume $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$ to be an instance of the VSSP-PM problem and assume r to be the maximum ratio of the largest component to the smallest component of the d-dimensional vectors among all jobs, i.e.,

$$r = \max_{J_j \in J} \left\{\frac{\max_k p_j^k}{\min_k p_j^k}\right\}. \tag{4}$$

Note that when $r \geq 1$, in particular, we set $r = +\infty$ if $\min_{J_j \in J}\left\{\min_k p_j^k\right\} = 0$.

Now, we construct an auxiliary instance $I' = (J', M', p'(\cdot), \pi'(\cdot))$ for an instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$ of VSSP-PM problem, where $J' = \{J_1', J_2', \ldots, J_n'\}$ is a set of n jobs, $M' = \{M_1', M_2', \ldots, M_m'\}$ is a set of m parallel machines, and for each $J_j' \in J'$, we define

$$p_j = \begin{cases} \min\limits_{i\in[d]} p_j^i, & \text{if } r \le d, \\ \sum\limits_{i=1}^{d} p_j^i, & \text{otherwise,} \end{cases} \tag{5}$$

and $p'(J_j') = p_j, \pi'(\cdot) = \pi(\cdot)$, i.e., for each subset $\ell \subseteq [n] := \{1, 2, \ldots, n\}, \pi'(\cup_{k\in\ell}\{J_k'\}) = \pi(\cup_{k\in\ell}\{J_k\})$. We call this instance $I' = (J', M', p'(\cdot), \pi'(\cdot))$ as an auxiliary instance of $I = (J, M, p(\cdot), \pi(\cdot))$. Then, we have the following result.

Lemma 2. *Given an instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$ of the VSSP-PM problem and its auxiliary instance $I' = (J', M', p'(\cdot), \pi'(\cdot))$, suppose that $\sigma' = (A_1', \ldots, A_m', R')$ is a feasible solution of the instance I', then there exists a feasible solution $\sigma = (A_1, \ldots, A_m, R)$ of instance I satisfying*

$$\begin{cases} \max\limits_{i\in[m]} \left\{\left\|\mathbf{p}(A_i)\right\|_\infty\right\} + \pi(R) \le r\left(\max\limits_{i\in[m]} \left\{\sum\limits_{J_j'\in A_i'} p_j\right\} + \pi'\left(R'\right)\right), & \text{if } r \le d, \\ \max\limits_{i\in[m]} \left\{\left\|\mathbf{p}(A_i)\right\|_\infty\right\} + \pi(R) \le \max\limits_{i\in[m]} \left\{\sum\limits_{J_j'\in A_i'} p_j\right\} + \pi'\left(R'\right), & \text{otherwise,} \end{cases} \tag{6}$$

where $\mathbf{p}(A_i) = \sum_{J_j\in A_i}\mathbf{p}_j := (\sum_{J_j\in A_i} p_j^1, \sum_{J_j\in A_i} p_j^2, \ldots, \sum_{J_j\in A_i} p_j^d)$.

Proof. We are given an instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$ of the VSSP-PM problem and its auxiliary instance $I' = (J', M', p'(\cdot), \pi'(\cdot))$. Let $\sigma' = (A_1', \cdots, A_m', R')$ be a feasible solution of the instance I', and let $R = \{J_j \in J \mid J_j' \in R'\}$ and $A_i = \{J_j \in J_j' \mid \in A_i'\}$ for each $i \in [m]$. It is easy to obtain that $\sigma = (A_1, \cdots, A_m, R)$ is a feasible solution of instance I. We define $\mathbf{p}(A_i) = \sum_{J_j\in A_i}\mathbf{p}_j := (\sum_{J_j\in A_i} p_j^1, \sum_{J_j\in A_i} p_j^2, \cdots, \sum_{J_j\in A_i} p_j^d)$ as the load vector of machine M_i for each $i \in [m]$.

Case 1: $r \le d$.

In this case, we have $p_j = \min_{k\in[d]} p_j^k$ for each $j \in [n]$ and we can obtain the following for each $i \in [m]$:

$$\begin{aligned} p(A_i') &= \sum_{J_j'\in A_i'} p_j \\ &= \sum_{J_j\in A_i} \min_{k\in[d]} p_j^k, \\ \left\|\mathbf{p}(A_i)\right\|_\infty &= \max_{k\in[d]} \left\{\sum_{J_j\in A_i} p_j^k\right\} \le \sum_{J_j\in A_i} \max_{k\in[d]} p_j^k \\ &= \sum_{J_j\in A_i} \min_{k\in[d]} p_j^k \cdot \frac{\max_{k\in[d]} p_j^k}{\min_{k\in[d]} p_j^k} \\ &\le \sum_{J_j\in A_i} r \cdot \min_{k\in[d]} p_j^k \\ &= r \cdot \sum_{J_j'\in A_i'} p_j \\ &= r \cdot p(A_i'). \end{aligned} \tag{7}$$

Therefore, we can obtain the following equation:

$$\max_{i\in[m]} \left\{\left\|\mathbf{p}(A_i)\right\|_\infty\right\} + \pi(R) \le r \cdot \left(\max_{i\in[m]} \left\{\sum_{J_j'\in A_i'} p_j\right\} + \pi'\left(R'\right)\right). \tag{8}$$

Case 2: $r > d$.

In this case, we can obtain the following for each $i \in [m]$:

$$\begin{aligned} \left\|\mathbf{p}(A_i)\right\|_\infty &\le \left\|\mathbf{p}(A_i)\right\|_1 \\ &= \sum_{J_j\in A_i} p_j \\ &= \sum_{J_j'\in A_i'} p'\left(J_j'\right), \end{aligned} \tag{9}$$

implying

$$\begin{aligned} \arg\max_{i\in[m]} \left\{\left\|\mathbf{p}(A_i)\right\|_\infty\right\} &= \left\|\mathbf{p}(A_{i'})\right\|_\infty \\ &\le \left\|\mathbf{p}(A_{i'})\right\|_1 \\ &\le \max_{i\in[m]} \left\{\left\|\mathbf{p}(A_i')\right\|_1\right\} \\ &= \max_{i\in[m]} \left\{\sum_{J_j'\in A_i'} p_j\right\}. \end{aligned} \tag{10}$$

Thus, we have

$$\max_{i\in[m]} \left\{\left\|\mathbf{p}(A_i)\right\|_\infty\right\} + \pi(R) \le \max_{i\in[m]} \left\{\sum_{J_j'\in A_i'} p_j\right\} + \pi'\left(R'\right). \tag{11}$$

To sum up, we conclude this lemma. □

3. Approximation Algorithms for the VSSP-PM Problem

In this section, we will present a combination $(2-1/m)(\min\{r,d\})$-approximation algorithm to solve the VSSP-PM problem. Now, we first recall some results in reference [17] for the parallel machine scheduling with submodular penalties.

3.1. Overview of the Algorithm in Reference [17]. Liu and Li [17] proposed an algorithm to solve the parallel machine scheduling problem with submodular penalties, utilizing a combination of strategies from the greedy method and the List Scheduling (LS) algorithm. This algorithm consists of three phases.

In the first phase, we construct a set of jobs, denoted as $R(j')$, for each $p_{j'}$ in the sequences $p_0, p_1, p_2, \cdots, p_n$. This set contains jobs J_j for which $p_j > p_{j'}$, with $p_0 = 0$.

In the second phase, we construct the set $R^{j'}$ by using the method from [16]. $R^{j'}$ is a solution of the optimization problem $\min\_\{S: R(j') \subseteq S \subseteq J\}\ \{1/m\sum_{J_j \in J \setminus S} p_j + \pi(S)\}$. The LS algorithm is then applied to assign the remaining jobs in $A^{j'}$ $(= A \backslash R^{j'})$ to the m parallel machines.

In the last phase, we select the best one among all the feasible solutions obtained in the previous steps.

Formally, the algorithm can be described as follows.

For the Algorithm 1, they have the following result.

Lemma 3 (see [17]). *Algorithm 1 is a combinatorial $(2-1/m)$-approximation algorithm for the parallel machine scheduling with submodular penalties.*

3.2. VSSP-PM Problem. Taking inspiration from the algorithm presented in [17] and Lemma 2, we propose our GLS algorithm to solve the VSSP-PM problem. Our algorithm involves the following strategies: (1) Constructing an auxiliary instance $I' = (J', M', p'(\cdot), \pi'(\cdot))$ for a given instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$ of the VSSP-PM problem. (2) Utilizing Algorithm 1 to determine a feasible solution $(A_1', \ldots, A_m', R')$ for the instance I'. (3) Constructing a feasible solution $\sigma = (A_1, \ldots, A_m, R)$ for the instance I based on $(A_1', \ldots, A_m', R')$. The formal description of the algorithm is as follows.

Note that Algorithm 2 is a generalization of Algorithm 1. Our algorithm can handle the d-dimensional vector scheduling with submodular penalties on parallel machines, where $d \geq 1$. Using Algorithm 2, we can obtain the following result.

Theorem 4. *Algorithm 2 is $(2-1/m)(\min\{r,d\})$-approximation algorithm to solve the VSSP-PM problem.*

Proof. Given an instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$ of the VSSP-PM problem, we may assume that $\sigma^* = (A_1^*, \ldots, A_m^*, R^*)$ and $\sigma^* = (A_1^*, \ldots, A_m^*, R^*)$ are optimal solutions for the instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$ and $I' = (J', M', p'(\cdot), \pi'(\cdot))$, respectively.

Let $p_{j^*} = \max\{p_j \mid J_j' \in A^*\}$ and $J_{j^*}' \in A^*\ (= J' \setminus R^*)$. Then, we have

$$R(j^*) = \left\{J_j' \,\middle|\, j > j^*\right\} \subseteq R^{j^*}. \tag{12}$$

Let $(A_1^{j^*}, \ldots, A_m^{j^*}, R^{j^*})$ be the partition generated at Step 2 of the Algorithm 1 in the j^*-th loop. Because $R^{j^*} = \operatorname{argmin}_{S:\ R(j^*) \subseteq S \subseteq J'}\{w(S)\}$, we have $w(R^{j^*}) \leq w(R^*)$, where $w(S) = 1/m\sum_{J_j \in J' \setminus S} p_j + \pi(S)$. For each $S \subseteq J'$, we define $p(S) = \sum_{J_j' \in S} p_j$. Then, we can obtain the following equation:

$$\begin{aligned} \frac{1}{m}p\left(A^{j^*}\right) + \pi\left(R^{j^*}\right) &\leq \frac{1}{m}p\left(A^*\right) + \pi\left(R^*\right), \\ \pi\left(R^{j^*}\right) &\leq \frac{p\left(A^*\right) - p\left(A^{j^*}\right)}{m} + \pi\left(R^*\right), \end{aligned} \tag{13}$$

where $A^{j^*} = J' \setminus R^{j^*}$.

For convenience, we may assume that $p(A_{\widehat{i}}^{j^*}) = \arg\min_{i \in [m]}\left\{p(A_i^{j^*})\right\}$ and $J_{\widehat{j}}' \in J'$ is the last job added to $A_{\widehat{i}}^{j^*}$ by the LS algorithm [1]. Therefore, we have

$$\begin{aligned} p\left(A_{\widehat{i}}^{j^*}\right) &\leq \frac{p\left(J' \backslash R^{j^*}\right) - p_{\widehat{j}}}{m} + p_{\widehat{j}} \\ &= \frac{p\left(A^{j^*}\right)}{m} + \left(1 - \frac{1}{m}\right)p_{\widehat{j}} \\ &\leq \frac{p\left(A^{j^*}\right)}{m} + \left(1 - \frac{1}{m}\right)p_{j^*}, \end{aligned} \tag{14}$$

where the first inequality comes from the fact $p(J' \setminus R^{j^*}) - p_{\widehat{j}}/m$ is the average load on the m parallel machines, and the second inequality follows from the fact $J_{\widehat{j}}' \in A_{\widehat{i}}^{j^*}$ and $p_{\widehat{j}} \leq p_{j^*}$ for each $\widehat{j}\ (\leq j^*)$.

Additionally, we have

$$\max_{i \in [m]}\{p(A_i^*)\} \geq \frac{p(A^*)}{m} \text{ and } \max_{i \in [m]}\{p(A_i^*)\} \geq p_{j^*} = p'\left(J_{j^*}'\right), \tag{15}$$

where $(A_1^*, \ldots, A_m^*, R^*)$ is the optimal partition of the instance $I' = (J', M', p'(\cdot), \pi'(\cdot))$ and $A^{j^*} = J' \setminus R^{j^*}$, $J_{j^*}' \in A^{j^*}$.

Suppose that $(A_1', \ldots, A_m', R')$ is the minimum partition among $\left\{(A_1^{j'}, \ldots, A_m^{j'}, R^{j'}) \mid j' \in [n] \cup \{0\}\right\}$ and $(A_1, \ldots, A_m, R)$ is a feasible solution of instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$, where $R = \left\{J_j \in J \mid J_j' \in R'\right\}$ and for each $i \in [m]$, $A_i = \left\{J_j \in J \mid J_j' \in A_i'\right\}$.

By Lemma 2, we consider the following two cases.

Case 1: $r \leq d$. We can obtain the following for each $i \in [m]$

Input: An instance $I = (J, M, p(\cdot), \pi(\cdot))$ of the parallel machine scheduling with submodular penalties.
Output: A feasible solution $\sigma = (A_1, \cdots, A_m, R)$ of the instance I.
(1) Let $p_0 \coloneqq 0$ and sort all jobs in J in nondecreasing order, for convenience, we may assume that $p_0 \le p_1 \le p_2 \le \cdots \le p_n$.
(2) For $j' = 0, 1, \ldots, n$ do
Constructing $R(j') = \{J_j \mid j > j'\}$ and finding the set $R^{j'} = \operatorname{argmin}_{S:\, R(j') \subseteq S \subseteq J} \left\{1/m \sum_{J_j \in J \setminus S} p_j + \pi(S)\right\}$ by the method in [16]. We assign the jobs in $A^{j'}$ $(= A \setminus R^{j'})$ to $A_1^{j'}, A_2^{j'}, \ldots, A_m^{j'}$ using the LS algorithm.
(3) Let $\overline{j} = \arg \min_{j'} \left\{\max_i \left\{\sum_{p_j \in A_i^{j'}} p_j\right\} + \pi(R^{j'})\right\}, R \coloneqq R^{\overline{j}}, A_i \coloneqq A_i^{\overline{j}}$ for each $i = 1, 2, \ldots, m$.
(4) Output the feasible solution $\sigma = (A_1, \ldots, A_m, R)$ of instance I.

Algorithm 1: Algorithm based on the greedy and LS algorithms.

Input: An instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$ of the VSSP-PM problem.
Output: A feasible solution $\sigma = (A_1, \ldots, A_m, R)$ of the instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$.
(1) Constructing the auxiliary instance $I' = (J', M', p'(\cdot), \pi'(\cdot))$ of $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$.
(2) Using the algorithm 1 to find a feasible solution $(A_1', \ldots, A_m', R')$ of the instance I', and construct a feasible solution $\sigma = (A_1, \ldots, A_m, R)$ of instance I, where $R = \{J_j \in J \mid J_j' \in R'\}$ and for each $i \in [m], A_i = \{J_j \in J \mid J_j' \in A_i'\}$.
(3) Output the feasible schedule $\sigma = (A_1, \ldots, A_m, R)$ of instance I.

Algorithm 2: GLS algorithm.

$$
\begin{aligned}
\max_{i \in [m]} \left\{\left\|\mathbf{p}(A_i)\right\|_\infty\right\} + \pi(R) &\le r \cdot \left(\max_{i \in [m]} \left\{\sum_{J_j' \notin A_i'} p_j\right\} + \pi'\left(R'\right)\right) \\
&= r \cdot \left(\max_{i \in [m]} \left\{p\left(A_i'\right)\right\} + \pi'\left(R'\right)\right) \\
&\le r \cdot \left(p\left(A_i^{j^*}\right) + \pi\left(R^{j^*}\right)\right) \\
&\le r \cdot \left(\frac{p\left(A^{j^*}\right)}{m} + \left(1 - \frac{1}{m}\right) p_{j^*} + \pi\left(R^{j^*}\right)\right) \\
&\le r \cdot \left(\frac{p\left(A^{j^*}\right)}{m} + \left(1 - \frac{1}{m}\right) p_{j^*} + \frac{p(A^*) - p\left(A^{j^*}\right)}{m} + \pi(R^*)\right) \\
&\le r \cdot \left(\frac{p(A^*)}{m} + \pi(R^*) + \left(1 - \frac{1}{m}\right) p_{j^*}\right) \\
&\le \left(2 - \frac{1}{m}\right) r \cdot \left(\max_{i \in [m]} \left\{p\left(A_i^*\right)\right\} + \pi(R^*)\right).
\end{aligned}
\tag{16}
$$

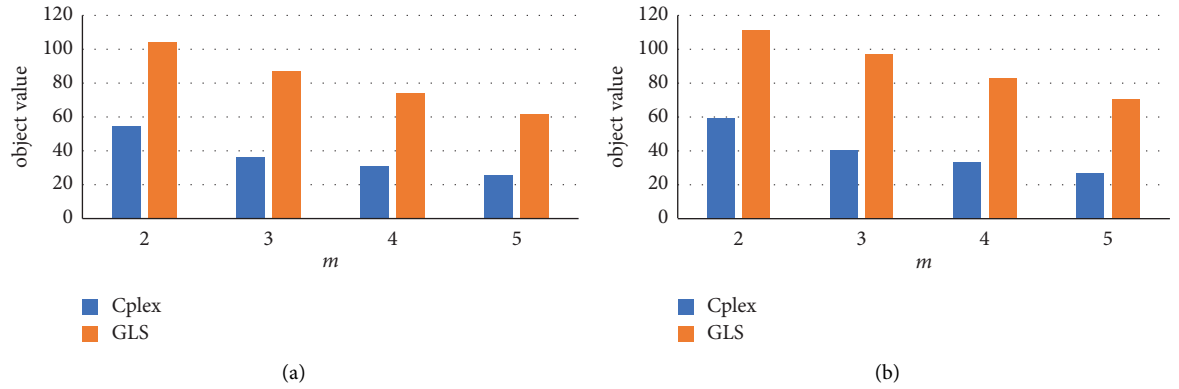

Figure 1: The comparison of two methods for the VSSP-PM problem. (a) $n = 10$, $d = 3$ and (b) $n = 10$, $d = 5$.

For convenience, let $(A_1'', \ldots, A_m'', R'')$ be another feasible solution of instance $I' = (J', M', p'(\cdot), \pi'(\cdot))$, where $R'' = \{J_j' \in J' \mid J_j \in R^*\}$ and for each $i \in [m]$, $A_i'' = \{J_j' \in J' \mid J_j \in A_i^*\}$. We may note that $p_j = \min_{k\in[d]} p_j^k$ for each $j \in [n]$. Then, we have

$$\begin{aligned}\max_{i\in[m]}\{p(A_i^*)\} + \pi(R^*) &\le \max_{i\in[m]}\{p(A_i'')\} + \pi\left(R''\right)\\ &= \arg\max_{i\in[m]}\left\{\sum_{J_j'\in A_i''} p_j\right\} + \pi(R^*)\\ &= \sum_{J_j'\in A_{i_t}} p_j + \pi(R^*)\\ &= \sum_{J_j'\in A_{i_t}} \min_{k\in[d]} p_j^k + \pi(R^*)\\ &\le \left\|p\left(A_{i_t}^*\right)\right\|_\infty + \pi(R^*)\\ &\le \max_{i\in[m]}\left\{\left\|p(A_i^*)\right\|_\infty\right\} + \pi(R^*) \le \mathrm{OPT},\end{aligned} \tag{17}$$

where $\mathrm{OPT} = \max_{i\in[m]}\{p(A_i^*)\} + \pi(R^*)$ is the objective value of the optimal solution $(A_1^*, \ldots, A_m^*, R^*)$ of the instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$.

Thus, we have

$$\begin{aligned}\max_{i\in[m]}\left\{\left\|\mathbf{p}(A_i)\right\|_\infty\right\} + \pi(R) &\le \left(2 - \frac{1}{m}\right) r \cdot \mathrm{OPT}\\ &\le \left(2 - \frac{1}{m}\right)\min\{r, d\} \cdot \mathrm{OPT}.\end{aligned} \tag{18}$$

Case 2: $r > d$. Using similar arguments as Case 1, we can obtain the following for each $i \in [m]$

$$\begin{aligned}\max_{i\in[m]}\left\{\left\|\mathbf{p}(A_i)\right\|_\infty\right\} + \pi(R) &\le \max_{i\in[m]}\left\{\sum_{J_j'\in A_i'} p_j\right\} + \pi'\left(R'\right)\\ &= \max_{i\in[m]}\{p(A_i')\} + \pi'\left(R'\right)\\ &\le \left(2 - \frac{1}{m}\right)\max_{i\in[m]}\{p(A_i^*)\} + \pi(R^*).\end{aligned} \tag{19}$$

Similarly, let $(A_1'', \ldots, A_m'', R'')$ be the solution of instance $I' = (J', M', p'(\cdot), \pi'(\cdot))$, where $R'' = J_j' \in J' \mid J_j \in R^*$ and for each $i \in [m]$, $A_i'' = \{J_j' \in J' \mid J_j \in A_i^*\}$. Then, we have

$$\begin{aligned}\max_{i\in[m]}\{p(A_i^*)\} + \pi(R^*) &\le \max_{i\in[m]}\{p(A_i'')\} + \pi\left(R''\right)\\ &= \arg\max_{i\in[m]}\left\{\left\|A_i^*\right\|_1\right\} + \pi(R^*)\\ &= \left\|A_{i_k}^*\right\|_1 + \pi(R^*)\\ &\le d \cdot \left\|A_{i_k}^*\right\|_\infty + \pi(R^*)\\ &\le d \cdot \max_{i\in[m]}\left\{\left\|A_i^*\right\|_\infty\right\} + \pi(R^*)\\ &\le d \cdot \mathrm{OPT},\end{aligned} \tag{20}$$

where $\mathrm{OPT} = \max_{i\in[m]}\{p(A_i^*)\} + \pi(R^*)$ is the objective value of the optimal solution $(A_1^*, \ldots, A_m^*, R^*)$ of the instance $I = (J, M, \mathbf{p}(\cdot), \pi(\cdot))$.

Therefore, we have

$$\begin{aligned}\max_{i\in[m]}\left\{\left\|\mathbf{p}(A_i)\right\|_\infty\right\} + \pi(R) &\le \left(2 - \frac{1}{m}\right) d \cdot \mathrm{OPT}\\ &\le \left(2 - \frac{1}{m}\right)\min\{r, d\} \cdot \mathrm{OPT}.\end{aligned} \tag{21}$$

To sum up, we reach the conclusion.

Considering the function $\pi(S) = \sum_{J_j \in S} \pi(J_j) - \theta|S|^2$, where $S(\neq \varnothing) \subseteq J$ and $\pi(\varnothing) = 0$. Here, $\pi(J_j) > 0$ for each $J_j \in J$, and $\theta \geq 0$ is a given constant. As θ approaches zero, the $\pi(\cdot)$ becomes increasingly approximated as a linear function. It can be easily verified that $\pi(\cdot)$ is a submodular function.

The numerical experiments of our algorithm are described as follows. The experimental data are generated in an average distribution over a given range (Dataset link: https://github.com/wencheng2018/GLSalgorithm-based-data), so each experiment is repeated 10 times. The final results are averaged to reduce the impact of randomness.

We compared the GLS algorithm and the optimal solution for the VSSP-PM problem. We use IBM's open-source tool, CPLEX, to obtain the optimal solution. In cases where the optimal solution was not obtained within 1 minute, we terminated the CPLEX process. As illustrated in Figure 1, the results obtained from the GLS algorithm closely approximate the optimal solution. The numerical experiment results indicate that the approximation ratio of our algorithm is less than 2.5 for the given instances. This finding indicates the superiority of our method.

4. Conclusions

In this paper, we proposed the problem of vector scheduling with submodular penalties on parallel machines (VSSP-PM), which generalizes both the vector scheduling with submodular rejection on a single machine [13] and the parallel machine scheduling with submodular penalties [17]. We present a $(2 - 1/m)(\min\{r, d\})$-approximation algorithm, where m is the number of parallel machines and r is the maximum ratio of the largest component to the smallest component of the d-dimension vectors among all jobs. This result generalizes the conclusions in [17, 20]. Numerical experiment results show that our algorithm is efficient and reliable.

A challenging task for further research is to present some better approximation algorithms with lower performance or lower running times to solve the VSSP-PM problem. Additionally, vector scheduling with submodular penalties on unrelated machines is an interesting problem to be explored. It is possible to design a $O(d)$-approximation algorithm, but it is a challenge.

Data Availability

All relevant data used to support the findings of this study are within the paper.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

The work was supported by the Yunnan Minzu University.

References

[1] R. L. Graham, "Bounds for certain multiprocessing anomalies," *Bell System Technical Journal*, vol. 45, no. 9, pp. 1563–1581, 1966.

[2] J. K. Lenstra, D. B. Shmoys, and É. Tardos, "Approximation algorithms for scheduling unrelated parallel machines," *Mathematical Programming*, vol. 46, no. 1-3, pp. 259–271, 1990.

[3] X. Chen, Y. Liang, M. Sterna, W. Wang, and J. Błażewicz, "Fully polynomial time approximation scheme to maximize early work on parallel machines with common due date," *European Journal of Operational Research*, vol. 284, no. 1, pp. 67–74, 2020.

[4] M. R. Garey and D. S. Johnson, "Computers and intractability: a guide to the theory of NP-completeness," *A Series of Books in the Mathematical Sciences*, Freeman, New York, NY, USA, 1979.

[5] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM Journal on Applied Mathematics*, vol. 17, no. 2, pp. 416–429, 1969.

[6] K. Jansen, K. M. Klein, and J. Verschae, "Closing the gap for makespan scheduling via sparsification techniques," *Mathematics of Operations Research*, vol. 45, no. 4, pp. 1371–1392, 2020.

[7] Y. Bartal, S. Leonardi, A. Marchetti-Spaccamela, J. Sgall, and L. Stougie, "Multiprocessor scheduling with rejection," *SIAM Journal on Discrete Mathematics*, vol. 13, no. 1, pp. 64–78, 2000.

[8] J. Ou, X. Zhong, and G. Wang, "An improved heuristic for parallel machine scheduling with rejection," *European Journal of Operational Research*, vol. 241, no. 3, pp. 653–661, 2015.

[9] C. Chekuri and S. Khanna, "On multidimensional packing problems," *SIAM Journal on Computing*, vol. 33, no. 4, pp. 837–851, 2004.

[10] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, Cambridge, UK, 1995.

[11] D. G. Harris and A. Srinivasan, "The Moser-Tardos framework with partial resampling," *Journal of the ACM*, vol. 66, no. 5, pp. 1–45, 2019.

[12] B. Dai and W. Li, "Vector scheduling with rejection on two machines," *International Journal of Computer Mathematics*, vol. 97, no. 12, pp. 2507–2515, 2020.

[13] W. Li and Q. Cui, "Vector scheduling with rejection on a single machine," *4OR-Q Journal of Operational Research*, vol. 16, no. 1, pp. 95–104, 2018.

[14] N. Bansal, T. Oosterwijk, T. Vredeveld, and R. van der Zwaan, "Approximating vector scheduling: almost matching upper and lower bounds," *Algorithmica*, vol. 76, no. 4, pp. 1077–1096, 2016.

[15] L. Lovász, *Submodular Functions and Convexity*, Mathematical Programming the State of the Art, Bonn, Germany, 1982.

[16] L. Fleischer and S. Iwata, "A push-relabel framework for submodular function minimization and applications to parametric optimization," *Discrete Applied Mathematics*, vol. 131, no. 2, pp. 311–322, 2003.

[17] X. Liu and W. Li, "Approximation algorithms for the multiprocessor scheduling with submodular penalties," *Optics Letters*, vol. 15, no. 6, pp. 2165–2180, 2021.

[18] W. Wang and X. Liu, "A combinatorial 2-approximation algorithm for the parallel-machine scheduling with release times and submodular penalties," *Mathematics*, vol. 10, no. 1, p. 61, 2021.
[19] X. Zhang, D. Xu, D. Du, and C. Wu, "Approximation algorithms for precedence-constrained identical machine scheduling with rejection," *Journal of Combinatorial Optimization*, vol. 35, no. 1, pp. 318–330, 2018.
[20] X. Liu, W. Li, and Y. Zhu, "Single machine vector scheduling with general penalties," *Mathematics*, vol. 9, no. 16, p. 1965, 2021.