

APPROXIMATION ALGORITHMS FOR THE INTEGRATED PATH AND BIN PACKING PROBLEM

WEIDONG LI^{1,2,*}  AND RUIQING SUN¹

Abstract. In this paper, we consider the integrated path and bin packing problem, which to use the minimum number of unit-size bins to packing the arcs on the path between two specific vertices in a given directed graph. We propose a $3/2$ -approximation algorithm and an asymptotic polynomial time approximation scheme.

Mathematics Subject Classification. 68W25, 68W40.

Received August 30, 2023. Accepted December 28, 2024.

1. INTRODUCTION

Traditional network optimization problems aim at constructing networks with good performance. The integrated network design and scheduling (INDS) problems are a class of operations that change the characteristics of a network through selection and scheduling. For example, after a massive damage to an infrastructure network caused by an extreme event (*e.g.*, hurricane, earthquake), infrastructure managers must develop a recovery plan to repair damaged components so that the performance of the network is restored to a certain level. For customers of infrastructure networks, the success of recovery plans will depend on how quickly their services are restored. Therefore, damaged parts are repaired and operational networks are constructed to guarantee a quick return to service. This is a must for many customers. However, the INDS model can help infrastructure managers develop remediation plans and assign infrastructure staff to repair infrastructure networks. More applications of INDS include the implementation of network upgrades to improve the services provided by the network and the creation and construction of a new network, which is often needed in humanitarian logistics activities (see, *e.g.*, [13–15]).

Nurre and Sharkey [13] introduced the INDS problem on parallel identical machines, which is to select a set of nodes and arcs for restoration and then schedule them on a set of identical parallel machines (or work groups) such that makespan is minimized under the performance level constraint, where the performance of the level constraint is evaluated by solving classic network optimization problems. They proposed a new heuristic scheduling rule algorithm framework that selects and schedules arc sets based on their interactions in the network. If the number of machines is one, the INDS problem is exactly the classical network design problem. If performance is determined by a path with distance no more than a given upper bound, the INDS problem

Keywords. Bin packing, shortest path, approximation algorithm, asymptotic polynomial time approximation scheme.

¹ School of Mathematics and Statistics, Yunnan University, Kunming 650504, P.R. China.

² Dianchi College of Yunnan University, Kunming 650228, P.R. China.

*Corresponding author: weidongmath@126.com

is exactly the restricted shortest path problem [7, 18]. If performance is determined by a spanning tree with distance no more than a given upper bound, the INDS problem is exactly the constrained minimum spanning tree problem [8]. If performance is determined by a matching with weight at least a given lower bound, the INDS problem is exactly the budgeted matching problem [2]. If performance is determined by a flow with value at least a given lower bound, the INDS problem is related to the problem of budget-constrained minimum cost flows [10, 11].

We use the following notations about approximation algorithms for a minimization problem. For an algorithm \mathcal{A} of a given problem, we denote its cost by \mathcal{A} as well. The cost of an optimal solution for the same problem is denoted by OPT . We define the asymptotic approximation ratio of an algorithm \mathcal{A} as the infimum $\rho \geq 1$ such that any input satisfies $\mathcal{A} \leq \rho \cdot \text{OPT} + O(1)$. The absolute approximation ratio of an algorithm \mathcal{A} is the infimum $\rho \geq 1$ such that for any input, $\mathcal{A} \leq \rho \cdot \text{OPT}$. An asymptotic polynomial time approximation scheme (APTAS) is a family of approximation algorithms such that for every $\epsilon > 0$ the family contains a polynomial time algorithm with an asymptotic approximation ratio of $1 + \epsilon$. An asymptotic fully polynomial time approximation scheme (AFPTAS) is an APTAS whose time complexity is polynomial not only in the input size but also in $\frac{1}{\epsilon}$. Polynomial time approximation schemes (PTAS) and fully polynomial time approximation schemes (FPTAS), are defined similarly, but are required to give an approximation ratio of $1 + \epsilon$ according to the absolute approximation ratio.

As mentioned in [15], no approximation algorithm with a nontrivial bound is known for the INDS problem and its special cases with specific performance functions. Saito and Shioura [15] proved that the INDS problem admits a PTAS when the number of machines is a constant, where the performance of the network is associated with some classic network optimization problems, such as the minimum spanning tree, shortest path, maximum flow with unit capacity, and maximum-weight matching.

The INDS problem is closely related to the combination of parallel machine scheduling and the covering problem which is first proposed in [16], where the performance level is unbounded, which means that the restored network only need to contain a specific structure. Epstein *et al.* [4] proposed a $(2 + \epsilon)$ -approximation algorithm for the combination of parallel machine scheduling and the vertex cover problem, which is to select a subset of the vertices such that for each edge at least one endpoint belongs to this subset and scheduling it on parallel machines so as to minimize the makespan. Wang *et al.* [17] proposed a unified approximation algorithm for the combination of parallel machine scheduling and the covering problem. Guan *et al.* [6] presented a $\frac{3}{2}$ -approximation algorithm and a PTAS for the combination problem of parallel machine scheduling and path.

Since the classical one-dimensional bin packing problem [5] and parallel machine scheduling [1, 9] are closely related, it is natural to consider the integrated network design and bin packing problem, which is to select a set of nodes and arcs for restoration and then pack them into a set of bins with limited capacity, such that the number of non-empty bins is minimized under the performance level constraint, as in [13]. Let's consider a scenario where the network is damaged due to a disaster such as an earthquake, flood, or hurricane. The manager hopes to repair the network as soon as possible to restore the performance of the network to a certain level. Then, the materials or equipment (also understood as goods) needed to repair the damaged components need to be dispatched. The goods to be shipped are usually packed into a number of boxes or cartons, usually the goods are indivisible and can fit into only one box at most. This requires us to pack the goods properly and minimize costs by using as few boxes as possible, such as trucks or cargo ships, to transport them to their destination. Specifically, Epstein *et al.* [4] proposed a $(2 + \epsilon)$ -approximation algorithm for the combination of bin packing and the vertex cover problem, which is to select a subset of the vertices such that for each edge at least one endpoint belongs to this subset and pack them into a set of bins with unit capacity so as to minimize the number of used bins. In this paper, we propose the integrated path and bin packing (IPBP) problem, which to use the minimum number of unit-size bins to packing the arcs on the path between two specific vertices in a given directed graph, where the performance is determined by a path with distance no more than a given upper bound. If the graph contains only a path, the IPBP problem is exactly the classical one-dimensional bin packing problem, which admits an APTAS [5] and an AFPTAS [12].

The rest of the paper is organized as follows. In Section 2, we introduce some notations about the IPBP problem. In Section 3, we present a simple $\frac{3}{2}$ -approximation algorithm. In Section 4, we design an APTAS by using a grouping technique. We present some conclusions and possible future research in the last section.

2. PRELIMINARIES

We are given an infinite number of unit-size bins B_i and a directed graph $D = (V, A; u, v; l; L)$ with two specific vertices $u, v \in V$, where $l : A \rightarrow (0, 1]$ is a length function, and L is a given upper bound on the length of the path from u to v . Without loss of generality, we assume that the arcs $a_j \in A$ are labelled in non-increasing order of the length, *i.e.*,

$$1 \geq l(a_1) \geq l(a_2) \geq \dots \geq l(a_{|A|}) > 0.$$

For convenience, let $l(B_i)$ denote the total length of the arcs packed in a bin B_i , where we will often identify bins with arcs assigned to them. The integrated path and bin packing (IPBP) problem is to find a directed path P_{uv} from u to v in D such that

$$l(P_{uv}) = l(A(P_{uv})) = \sum_{a \in A(P_{uv})} l(a) \leq L,$$

where $A(P_{uv})$ is the set of arcs in P_{uv} and the minimum number of bins needed to pack the arcs in the set $A(P_{uv})$ of arcs in the path P_{uv} , without violating the bin capacity 1.

As mentioned in [18], we can assume that $D = (V, A; u, v; l; L)$ is acyclic. This assumption simplifies the presentation, and extending the algorithm to general graphs is straightforward. For convenience, let $[n] = \{1, 2, \dots, n\}$ for any positive integer n , and $l(S) = \sum_{a \in S} l(a)$ for any subset $S \subseteq A$. Let P_{uv}^* be the u - v path and k^* be the number of used bins in the optimal solution. Clearly, we have

$$\text{OPT} = k^* \geq l(P_{uv}^*), \tag{1}$$

where OPT is the optimal value.

3. A $\frac{3}{2}$ -APPROXIMATION ALGORITHM

In this section, we design a $\frac{3}{2}$ -approximation algorithm, which generalizes the method in [6], where the length of the u - v path does not have an upper bound L . First, we divide the arcs into long arcs and short arcs. Next, for each possible number of long arcs in the optimal solution, we construct an integer linear program that can be solved optimally *via* dynamic programming. By using First-Fit-Decreasing (FFD) algorithm to pack the selected arcs, we obtain a feasible solution with an objective value of no more than $\frac{3}{2}$ OPT for the IPBP problem.

Let $\mathcal{L} = \{a_j \in A \mid \frac{1}{2} < l(a_j) \leq 1\}$ be the set of *long* arcs, and $A \setminus \mathcal{L}$ be the set of *short* arcs. For each arc $a_j \in A$, let

$$l_1(a_j) = \begin{cases} 1, & \text{if } a_j \in \mathcal{L}, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

For each $n_1 = 0, 1, 2, \dots, |A|$, we construct an integer linear program ILP(n_1) for restricted shortest path problem [7], where

$$\text{ILP}(n_1) : \begin{cases} \min l(P_{uv}) \\ l_1(P_{uv}) = n_1. \end{cases}$$

As mentioned in [6], the optimal u - v path $P_{uv}(n_1)$ for the above integer linear program ILP(n_1) can be found by using the standard dynamic programming method within time $O(n_1|A|)$ [7]. If $l(P_{uv}(n_1)) > L$, we discard $P_{uv}(n_1)$. Otherwise, put the arcs in $A(P_{uv}(n_1))$ into the bins by using the classical FFD algorithm [3]. Let $k(n_1)$

be the number of used bins in the solution produced by the FFD algorithm. We choose the best solution with the minimum value of $k(n_1)$ among at most $|A| + 1$ feasible solutions. Let k be the corresponding objective value.

In summary, our algorithm is as follows.

Algorithm 1.

Input: A directed acyclic graph $D = (V, A; u, v; l; L)$.

Output: A path P_{uv} from u to v and a packing of the arcs on P_{uv} .

Step 1. For each $n_1 = 0, 1, 2, \dots, |A|$, construct the integer linear program $\text{ILP}(n_1)$.

Step 2. Solve $\text{ILP}(n_1)$ optimally using dynamic programming in [7] and obtain an optimal path $P_{uv}(n_1)$.

Step 3. If $l(P_{uv}(n_1)) > L$, we discard $P_{uv}(n_1)$. Otherwise, put the arcs in $A(P_{uv}(n_1))$ into the bins by using the First-Fit-Decreasing (FFD) algorithm in [3].

Step 4. Output the path $P_{uv}(n_1)$ which uses the minimum number of bins and the corresponding packing.

We demonstrate an example to explain our problem. A visualization is shown in Figure 1. Given a directed graph $D = (V, A; u, v; l; L)$ with 5 nodes and 10 arcs such that $L = \frac{11}{5}$ and V, A as shown in Figure 1. Let $l(a_1) = 1/2, l(a_2) = 3/4, l(a_3) = 2/3, l(a_4) = 1/4, l(a_5) = 3/4, l(a_6) = 1/3, l(a_7) = 1/3, l(a_8) = 1/5, l(a_9) = 1/2$ and $l(a_{10}) = 1/4$, where $l(a_j)$ is the length of arc a_j for $j = 1, \dots, 10$. Next, we consider the construction and solution of $\text{ILP}(n_1)$. When $n_1 = 0, 4, 5, \dots, 10$, the corresponding restricted shortest path has no feasible solution, since there is no $u - v$ path with n_1 long arcs in graph G .

When $n_1 = 1$, the integer linear program $\text{ILP}(1)$ is

$$\text{ILP}(1) : \begin{cases} \min l(P_{uv}) \\ l_1(P_{uv}) = 1. \end{cases}$$

The corresponding optimal path is $P_{uv}(1) = \{u, a_4, v_3, a_5, v_4, a_6, v_5, a_7, v\}$, the red color path shown in Figure 1 and $l(P_{uv}(1)) < L$. Then, pack the arcs in $P_{uv}(1)$ into the bins by using the FFD algorithm and the number of used bins is 2.

When $n_1 = 2$, the integer linear program $\text{ILP}(2)$ is

$$\text{ILP}(2) : \begin{cases} \min l(P_{uv}) \\ l_1(P_{uv}) = 2. \end{cases}$$

The corresponding optimal path is $P_{uv}(2) = \{u, a_1, v_1, a_2, v_2, a_3, v\}$, the blue color path shown in Figure 1 and $l(P_{uv}(2)) < L$. Then, pack the arcs in $P_{uv}(2)$ into the bins by using the FFD algorithm and the number of used bins is 3.

When $n_1 = 3$, the integer linear program $\text{ILP}(3)$ is

$$\text{ILP}(3) : \begin{cases} \min l(P_{uv}) \\ l_1(P_{uv}) = 3. \end{cases}$$

The corresponding optimal path is $P_{uv}(3) = \{u, a_4, v_3, a_5, v_4, a_8, v_1, a_2, v_2, a_3, v\}$. However, $l(P_{uv}(3)) = l(A(P_{uv}(3))) > L$, we discard $P_{uv}(3)$. The specific packing is shown in Figure 1. Thus, output the path $P_{uv}(1)$ and the corresponding packing.

Next, we show that the algorithm proposed above is a $\frac{3}{2}$ -approximation algorithm.

Theorem 3.1. $k \leq \frac{3}{2}\text{OPT}$.

Proof. Let $n_1^* = |\mathcal{L} \cap A(P_{uv}^*)|$ be the number of long arcs on the optimal path P_{uv}^* . Since $l_1(P_{uv}^*) = |\mathcal{L} \cap A(P_{uv}^*)| = n_1^*$, P_{uv}^* is a feasible $u-v$ path for $\text{ILP}(n_1^*)$ with length no more than L . By the definitions of l_1 and n_1^* , we have

$$l(P_{uv}(n_1^*)) \leq l(P_{uv}^*) \leq L. \tag{3}$$

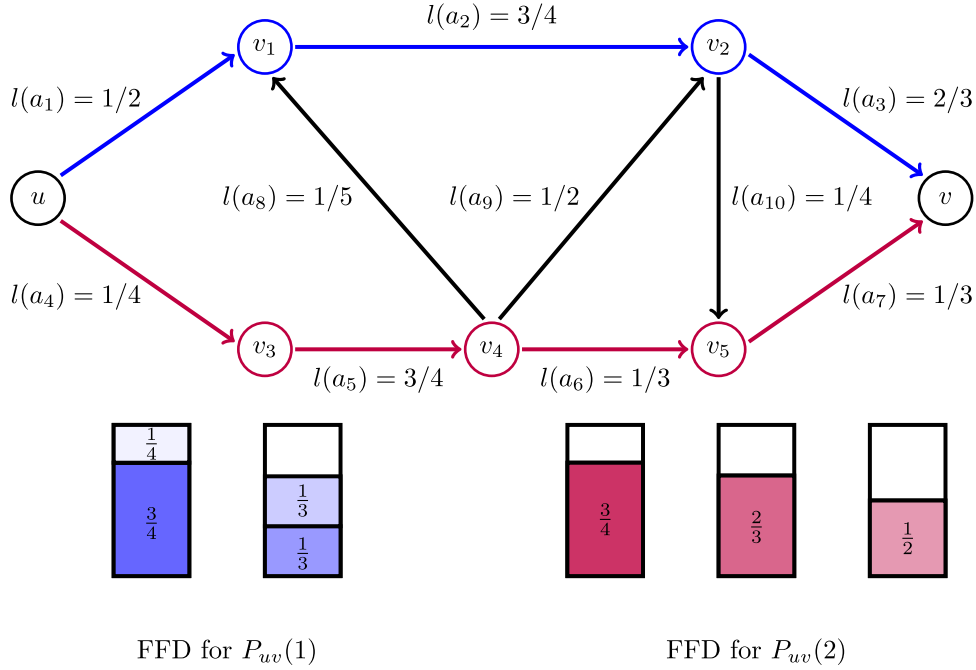


FIGURE 1. An example of Algorithm 1.

Consider the packing according to the FFD algorithm of the arcs in $A(P_{uv}(n_1^*))$. Assume that all the arcs are packed into the first $k(n_1^*)$ bins, where $k(n_1^*)$ is the number of used bins in FFD algorithm. Let $\kappa = \lceil \frac{2}{3}k(n_1^*) \rceil$. We distinguish the following two cases.

Case 1. The κ -th bin contains a long arc.

By the FFD algorithm, for $i \in [\kappa]$, each bin B_i contains a long arc. Therefore, we have

$$\text{OPT} = k^* \geq \kappa = \left\lceil \frac{2}{3}k(n_1^*) \right\rceil \geq \frac{2}{3}k(n_1^*).$$

Case 2. The κ -th bin does not contain a long arc.

According to the FFD algorithm, for $i = \kappa + 1, \kappa + 2, \dots, k(n_1^*)$, each bin B_i also does not contain any long arcs. Hence, the number of short arcs in $\cup_{i=\kappa}^{k(n_1^*)} B_i$ is at least

$$\begin{aligned} 2(k(n_1^*) - \kappa) + 1 &= 2\left(k(n_1^*) - \left\lceil \frac{2}{3}k(n_1^*) \right\rceil\right) + 1 \\ &\geq 2\left(k(n_1^*) - \frac{2}{3}k(n_1^*) - \frac{2}{3}\right) + 1 \\ &= \frac{2}{3}k(n_1^*) - \frac{1}{3} \\ &\geq \kappa - 1. \end{aligned}$$

Moreover, for each arc $a \in \cup_{i=\kappa}^{k(n_1^*)} B_i$, we have $l(a) + l(B_i) > 1$ for $i = 1, 2, \dots, \kappa - 1$. Therefore, $l(P_{uv}(n_1^*)) = \sum_{a \in \cup_{i=\kappa}^{k(n_1^*)} B_i} l(a) + \sum_{i=1}^{\kappa-1} l(B_i) > \kappa - 1$. It implies that

$$\text{OPT} = k^* \geq l(P_{uv}^*) \geq l(P_{uv}(n_1^*)) > \kappa - 1 = \left\lceil \frac{2}{3}k(n_1^*) \right\rceil - 1,$$

where the first inequality follows from (1) and the second inequality follows from (3). Therefore,

$$\text{OPT} = k^* \geq \left\lceil \frac{2}{3}k(n_1^*) \right\rceil \geq \frac{2}{3}k(n_1^*).$$

Since we choose the best solution with the minimum value of $k(n_1)$, we have $k \leq k(n_1^*) \leq \frac{3}{2}\text{OPT}$ in any case.

□

4. AN APTAS

In this section, we extend the method in the last section and design an APTAS for the IPBP problem. First, the arcs are divided into τ sets of long arcs (τ will be defined later) and a set of short arcs. Next, for each possible number of long arcs into τ sets in the optimal solution, we construct an integer linear program that can be solved optimally in polynomial time by using the method in [6]. Finally, packing the selected arcs into bins by using the method in [5, 12], we can obtain a feasible solution with an objective value of no more than $(1 + 7\epsilon)\text{OPT} + O(1)$ for the IPBP problem.

Given a desired accuracy $\epsilon \in (0, \frac{1}{2}]$ such that $\frac{1}{\epsilon}$ is an integer, let $\tau = \frac{1}{\epsilon^2}$. Let $A_0 = \{a \in A \mid 0 < l(a) \leq \epsilon\}$ and $A \setminus A_0$ be the set of *short* and *long* arcs, respectively. For each 2τ -dimensional vector $\mathbf{j} = (j_1^{\min}, j_1^{\max}, \dots, j_\tau^{\min}, j_\tau^{\max})$ with $1 \leq j_1^{\min} \leq j_1^{\max} < j_2^{\min} \leq \dots < j_\tau^{\min} \leq j_\tau^{\max} \leq |A|$, we construct τ mutually disjoint subsets $\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_\tau$ of $A \setminus A_0$, where

$$\mathcal{L}_t = \{a_j \in A \setminus A_0 \mid j_t^{\min} \leq j \leq j_t^{\max}\}, \text{ for each } t \in [\tau].$$

Given an arc $a \in A$, we construct a τ -dimensional vector $\mathbf{l}(a) = (l_1(a), l_2(a), \dots, l_\tau(a))$, which is defined as follows. If $a \in A_0$, let $l_t(a) = 0$ for $t = 1, 2, \dots, \tau$. If $a \in \mathcal{L}_t$, let $l_t(a) = 1$, and $l_{t'}(a) = 0$ for any $t' \neq t$. If $a \notin \cup_{t=1}^\tau \mathcal{L}_t \cup A_0$, let $l_t(a) = +\infty$ for each $t \in [\tau]$.

For each $n \in [|A|] \cup \{0\}$ and each vector $\mathbf{j} = (j_1^{\min}, j_1^{\max}, \dots, j_\tau^{\min}, j_\tau^{\max})$, we construct the following integer linear program $\text{ILP}(n, \mathbf{j})$:

$$\begin{cases} \min l(P_{uv}) \\ l_t(P_{uv}) = \lceil \epsilon^2 n \rceil, & \text{for } t = 1, \dots, h, \\ l_t(P_{uv}) = \lfloor \epsilon^2 n \rfloor, & \text{for } t = h + 1, \dots, \tau, \end{cases}$$

where $h = n - \frac{1}{\epsilon^2} \lfloor \epsilon^2 n \rfloor$. The following theorem states that $\text{ILP}(n, \mathbf{j})$ can be solved optimally in polynomial time.

Theorem 4.1 ([6]). *ILP(n, \mathbf{j}) can be solved optimally in $O(|A|^{O(\frac{1}{\epsilon^2})})$ time.*

Consider the set $A^* = A(P_{uv}^*)$ of arcs packed in the optimal solution. If $A^* \setminus A_0 = \emptyset$, let

$$\mathcal{L}_t^* = \emptyset, \text{ for each } t \in [\tau],$$

Otherwise, let $n^* = |A^* \setminus A_0|$ be the number of long arcs on the path P_{uv}^* and $h^* = n^* - \frac{1}{\epsilon^2} \lfloor \epsilon^2 n^* \rfloor$. We divide the set of long arcs in $A^* \setminus A_0$ into τ mutually disjoint subsets $A_1^*, A_2^*, \dots, A_\tau^*$ of long arcs, where

$$|A_t^*| = \lceil \epsilon^2 n^* \rceil \text{ for each } t \in [h^*], \text{ and } |A_t^*| = \lfloor \epsilon^2 n^* \rfloor, \text{ for } t = h^* + 1, \dots, \tau.$$

Moreover, for any two arcs $a_{j_1} \in A_{t_1}^*$ and $a_{j_2} \in A_{t_2}^*$ with $t_1 < t_2$, we have $j_1 < j_2$. Let $j_t^{\min,*}$ and $j_t^{\max,*}$ be the minimum and maximum indices of the arcs in A_t^* , respectively. For the 2τ -dimensional vector $\mathbf{j}^* =$

$(j_1^{\min,*}, j_1^{\max,*}, \dots, j_\tau^{\min,*}, j_\tau^{\max,*})$ with $1 \leq j_1^{\min,*} \leq j_1^{\max,*} < j_2^{\min,*} \leq \dots < j_\tau^{\min,*} \leq j_\tau^{\max,*} \leq |A|$, we construct τ mutually disjoint subsets $\mathcal{L}_1^*, \mathcal{L}_2^*, \dots, \mathcal{L}_\tau^*$ of $A \setminus A_0$, where

$$\mathcal{L}_t^* = \left\{ a_j \in A \setminus A_0 \mid j_t^{\min,*} \leq j \leq j_t^{\max,*} \right\}, \quad \text{for } t = 1, 2, \dots, \tau.$$

Since $n^*, j_t^{\min,*}, j_t^{\max,*} \in [|A|] \cup \{0\}$, the number of possibilities of $n^*, j_t^{\min,*}, j_t^{\max,*}$ is at most

$$(|A| + 1)^{2\tau+1} = |A|^{O(\frac{1}{\epsilon^2})}.$$

By trying all the possibilities, we can assume that the values of $n^*, j_t^{\min,*}, j_t^{\max,*}$ and the corresponding subsets \mathcal{L}_t^* are known.

Let $P_{uv}(n^*, \mathbf{j}^*)$ denote an optimal path for $\text{ILP}(n^*, \mathbf{j}^*)$, where $\mathbf{j}^* = (j_1^{\min,*}, j_1^{\max,*}, \dots, j_\tau^{\min,*}, j_\tau^{\max,*})$. Since

$$|A^* \cap \mathcal{L}_t^*| = |A_t^*| = \lceil \epsilon^2 n^* \rceil, \quad \text{for } t \in [h^*], \text{ and } |A^* \cap \mathcal{L}_t^*| = |A_t^*| = \lfloor \epsilon^2 n^* \rfloor,$$

for $t = h^* + 1, \dots, \tau$, P_{uv}^* is a feasible solution for $\text{ILP}(n^*, \mathbf{j}^*)$, implying that

$$l(P_{uv}(n^*, \mathbf{j}^*)) \leq l(P_{uv}^*) \leq L.$$

Let $A(n^*, \mathbf{j}^*) = A(P_{uv}(n^*, \mathbf{j}^*))$ be the set of arcs on the path $P_{uv}(n^*, \mathbf{j}^*)$. By the definitions, we have

$$|A(n^*, \mathbf{j}^*) \cap \mathcal{L}_t^*| = \lceil \epsilon^2 n^* \rceil, \quad \text{for } t \in [h^*], \text{ and } |A(n^*, \mathbf{j}^*) \cap \mathcal{L}_t^*| = \lfloor \epsilon^2 n^* \rfloor, \quad \text{for } t = h^* + 1, \dots, \tau.$$

The following lemma is the relationship between the minimum number of bins needed to pack the arcs in $A(n^*, \mathbf{j}^*)$ and the minimum number of bins needed to pack the arcs in the optimal path.

Lemma 4.2. $\text{OPT}(A(n^*, \mathbf{j}^*)) \leq (1+3\epsilon)\text{OPT}+2$, where $\text{OPT}(A(n^*, \mathbf{j}^*))$ is the minimum number of bins needed to pack the arcs in $A(n^*, \mathbf{j}^*)$.

Proof. In the optimal packing of A^* , assume that we only use the first $k^* = \text{OPT}$ bins B_i^* to pack the arcs in A^* , where B_i^* also denotes the set of arcs packed in bin B_i^* .

For each $i = 1, 2, \dots, k^*$, we pack $|B_i^* \cap \mathcal{L}_t^*|$ long arcs in $A(n^*, \mathbf{j}^*) \cap \mathcal{L}_{t+1}^*$ into bin B_i^* if there are, for $t = 1, 2, \dots, \tau - 1$. Then, put each long arc in $A(n^*, \mathbf{j}^*) \cap \mathcal{L}_1^*$ into a new bin. Finally, pack the short arcs in $A(n^*, \mathbf{j}^*) \cap A_0$ into the first k^* bins by using the FFD algorithm.

If all the short arcs can be packed into the first k^* bins, the number of used bins is at most

$$k^* + \lceil \epsilon^2 n^* \rceil \leq \text{OPT} + \epsilon \cdot \epsilon n^* + 1 \leq (1 + \epsilon)\text{OPT} + 1,$$

where the last inequality follows from that the total length of the arcs in $A^* \setminus A_0$ is at least $n^* \cdot \epsilon$ and (1).

If the short arc can not be packed in the first k^* bins, the number of used bins is at most

$$\begin{aligned} \frac{l(A(n^*, \mathbf{j}^*))}{1 - \epsilon} + 1 + \lceil \epsilon^2 n^* \rceil &\leq \frac{l(A(n^*, \mathbf{j}^*))}{1 - \epsilon} + \epsilon \cdot \epsilon n^* + 2 \\ &\leq \frac{l(A^*)}{1 - \epsilon} + \epsilon \text{OPT} + 2 \\ &\leq (1 + 3\epsilon)\text{OPT} + 2, \end{aligned}$$

where the last inequality follows from that $\epsilon \in (0, \frac{1}{2}]$ and (1). Hence, in any case, we have

$$\text{OPT}(A(n^*, \mathbf{j}^*)) \leq (1 + 3\epsilon)\text{OPT} + 2.$$

□

Finally, we pack the arcs in $A(n^*, \mathbf{j}^*)$. By using the method in [5, 12], we can find a feasible packing of the arcs in $A(n^*, \mathbf{j}^*)$ by using at most

$$\begin{aligned} (1 + \epsilon)\text{OPT}(A(n^*, \mathbf{j}^*)) + O(1) &\leq (1 + \epsilon)(1 + 3\epsilon)\text{OPT} + O(1) \\ &\leq (1 + 7\epsilon)\text{OPT} + O(1) \end{aligned} \quad (4)$$

bins.

In summary, our algorithm is as follows.

Algorithm 2.

Input: A directed acyclic graph $D = (V, A; u, v; l; L)$, and a desired accuracy $\epsilon \in (0, \frac{1}{2}]$.

Output: A path P_{uv} from u to v and a packing of the arcs on P_{uv} .

Step 1. For every pair (n, \mathbf{j}) , construct the integer linear program $\text{ILP}(n, \mathbf{j})$.

Step 2. Solve $\text{ILP}(n, \mathbf{j})$ optimally and obtain an optimal path $P_{uv}(n, \mathbf{j})$.

Step 3. If $l(P_{uv}(n, \mathbf{j})) \leq L$, use method in [5, 12] to find a feasible packing of the arcs on the path $P_{uv}(n, \mathbf{j})$. Otherwise, discard $P_{uv}(n, \mathbf{j})$.

Step 4. Output the path $P_{uv}(n, \mathbf{j})$ which uses the minimum number of bins and the corresponding packing.

Theorem 4.3. *Algorithm 2 is an APTAS for the IPBP problem.*

Proof. According to (4), the asymptotic approximation ratio of Algorithm 1 is $1 + 7\epsilon$. Since the number of possibilities of the pair (n, \mathbf{j}) is at most $|A|^{O(\frac{1}{\epsilon^2})}$, by Theorem 4.1, the total running time is

$$|A|^{O(\frac{1}{\epsilon^2})} \cdot |A|^{O(\frac{1}{\epsilon^2})} \cdot \text{poly}(|A|) = |A|^{O(\frac{1}{\epsilon^2})},$$

where $\text{poly}(|A|)$ is the running time of the method in [5, 12]. Therefore, the theorem holds. \square

5. CONCLUSION

We have proposed an absolute $\frac{3}{2}$ -approximation for the IPBP problem, which is a generalized version of the one-dimensional bin packing problem. Since the bin packing problem can not be approximated within $\frac{3}{2} - \epsilon$ for any $\epsilon > 0$, our algorithm could be the best result in the absolute sense, unless $P = NP$.

We have also presented an APTAS for the IPBP problem by using the method in [5, 12] and integer linear programming. It is well-known that the one-dimensional bin packing problem admits an AFPTAS [12]. Hence, it is very interesting to design an AFPTAS for the IPBP problem.

FUNDING

The work is supported in part by the National Natural Science Foundation of China [No. 12071417].

REFERENCES

- [1] N. Alon, Y. Azar, G.J. Woeginger and T. Yadid, Approximation schemes for scheduling on parallel machines. *J. Sched.* **1** (1998) 55–66.
- [2] A. Berger, V. Bonifaci, F. Grandoni and G. Schäfer, Budgeted matching and budgeted matroid intersection via the gasoline puzzle. *Math. Prog.* **128** (2011) 355–372.
- [3] G. Dósa, R. Li, X. Han and Z. Tuza, Tight absolute bound for First Fit Decreasing bin-packing: $\text{FFD}(L) \leq 11/9\text{OPT}(L) + 6/9$. *Theor. Comput. Sci.* **510** (2013) 13–61.
- [4] L. Epstein, A. Levin and G.J. Woeginger, Vertex cover meets scheduling. *Algorithmica* **74** (2016) 1148–1173.
- [5] W. Fernandez de La Vega and G.S. Lueker, Bin packing can be solved within $1 + \epsilon$ in linear time. *Combinatorica* **1** (1981) 349–355.

- [6] L. Guan, J. Li, W. Li and J. Lichen, Improved approximation algorithms for the combination problem of parallel machine scheduling and path. *J. Comb. Optim.* **38** (2019) 689–697.
- [7] R. Hassin, Approximation schemes for the restricted shortest path problems. *Math. Oper. Res.* **17** (1992) 36–42.
- [8] R. Hassin and A. Levin, An efficient polynomial time approximation scheme for the constrained minimum spanning tree problem using matroid intersection. *SIAM J. Comput.* **33** (2004) 261–268.
- [9] D.S. Hochbaum and D.B. Shmoys, Using dual approximation algorithms for scheduling problems theoretical and practical results. *J. ACM* **34** (1987) 144–162.
- [10] M. Holzhauser and S.O. Krumke, A generalized approximation framework for fractional network flow and packing problems. *Math. Methods Oper. Res.* **87** (2018) 19–50.
- [11] M. Holzhauser, S.O. Krumke and C. Thielen, On the complexity and approximability of budget-constrained minimum cost flows. *Inf. Process. Lett.* **126** (2017) 24–29.
- [12] N. Karmarkar and R.M. Karp, An efficient approximation scheme for the one-dimensional bin-packing problem, in Proceedings of 23rd Annual Symposium on Foundations of Computer Science (FOCS) (1982) 312–320.
- [13] S.G. Nurre and T.C. Sharkey, Integrated network design and scheduling problems with parallel identical machines: complexity and dispatching rules. *Networks* **63** (2014) 306–326.
- [14] S.G. Nurre, B. Cavdaroglu, J.E. Mitchell, T.C. Sharkey and W.A. Wallace, Restoring infrastructure systems: an integrated network design and scheduling (INDS) problem. *Eur. J. Oper. Res.* **223** (2012) 794–806.
- [15] Y. Saito and A. Shioura, Polynomial-time approximation schemes for a class of integrated network design and scheduling problems with parallel identical machines, in 7th International Symposium on Combinatorial Optimization (ISCO) (2022) 324–335.
- [16] Z. Wang and Z. Cui, Combination of parallel machine scheduling and vertex cover. *Theor. Comput. Sci.* **460** (2012) 10–15.
- [17] Z. Wang, W. Hong and D. He, A combination of parallel machine scheduling and the covering problem. *Pac. J. Optim.* **10** (2014) 577–591.
- [18] A. Warburton, Approximation of Pareto optima in multiple-objective shortest path problems. *Oper. Res.* **35** (1987) 70–79.



Please help to maintain this journal in open access!

This journal is currently published in open access under the Subscribe to Open model (S2O). We are thankful to our subscribers and supporters for making it possible to publish this journal in open access in the current year, free of charge for authors and readers.

Check with your library that it subscribes to the journal, or consider making a personal donation to the S2O programme by contacting subscribers@edpsciences.org.

More information, including a list of supporters and financial transparency reports, is available at <https://edpsciences.org/en/subscribe-to-open-s2o>.