MATHEMATICAL ENGINEERING TECHNICAL REPORTS

The Complexity of the Node Capacitated In-Tree Packing Problem

Shinji IMAHORI, Yuichiro MIYAMOTO,
Hideki HASHIMOTO, Yusuke KOBAYASHI,
Mihiro SASAKI, Mutsunori YAGIURA

METR 2009–29 July 2009

DEPARTMENT OF MATHEMATICAL INFORMATICS
GRADUATE SCHOOL OF INFORMATION SCIENCE AND TECHNOLOGY
THE UNIVERSITY OF TOKYO
BUNKYO-KU, TOKYO 113-8656, JAPAN

WWW page: http://www.keisu.t.u-tokyo.ac.jp/research/techrep/index.html

The Complexity of the Node Capacitated In-Tree Packing Problem

Shinji Imahori* Yuichiro Miyamoto⋆ Hideki Hashimoto◇
Yusuke Kobayashi* Mihiro Sasaki† Mutsunori Yagiura‡

*Department of Mathematical Informatics,
Graduate School of Information Science and Technology, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

⋆Department of Information and Communication Sciences,
Faculty of Science and Technology, Sophia University
7-1 Kioicho, Chiyoda-ku, Tokyo 102-8554, Japan

◇Department of Industrial and Systems Engineering,
Faculty of Science and Engineering, Chuo University
1-13-27 Kasuga, Bunkyo-ku, Tokyo 112-8551, Japan

†Department of Information Systems and Mathematical Sciences,
Faculty of Information Sciences and Engineering, Nanzan University
27 Seirei, Seto, Aichi 489-0863, Japan

‡Department of Computer Science and Mathematical Informatics,
Graduate School of Information Science, Nagoya University
Furocho, Chikusa-ku, Nagoya 464-8603, Japan

Abstract

This paper describes a node capacitated in-tree packing problem. The input consists of a directed graph, a root node, a node capacity function and edge consumption functions. The problem is to find the maximum number of rooted in-trees such that the total consumption of in-trees at each node does not exceed the capacity of the node. The problem is one of the network lifetime problems that are among the most important issues in the context of sensor networks. We reveal the computational complexity of the problem under various restrictions on consumption functions and graphs. For example, we consider general graphs, acyclic graphs and complete graphs embedded in the d-dimensional space $\mathbb{R}^d$ having edge consumption functions depending only on distances between end nodes.

Keywords: *Network design, computational complexity, wireless ad hoc network, sensor network.*

1 Introduction

In this paper, we consider a *node capacitated in-tree packing problem*. The input consists of a directed graph, a root node, a node capacity function and edge consumption functions. The problem is to find the maximum number of rooted in-trees such that the total consumption of in-trees at each node does not exceed the capacity of the node.

Formally, let $G = (V, E)$ be a directed graph. We call a subset $T \subseteq E$ *in-tree* if (V, T) is a directed spanning rooted in-tree (i.e., incoming arborescence). Let $\mathbb{R}_+$ be the set of nonnegative real numbers. Let

$h : E \to \mathbb{R}_+$ and $t : E \to \mathbb{R}_+$ be a head and a tail consumption function on directed edges, respectively. The consumption $c(T, v)$ of an in-tree T at a node $v \in V$ is defined as

$$c(T, v) := \sum_{e \in \delta_T^-(v)} h(e) + \sum_{e \in \delta_T^+(v)} t(e),$$

where $\delta_{E'}^-(v)$ (resp., $\delta_{E'}^+(v)$) is the set of edges in E' entering (resp., leaving) v. We call the first term of the above equation *head consumption*, and the second term *tail consumption*. Let $b : V \to \mathbb{R}_+$ be a node capacity function. The node capacitated in-tree packing problem is to find a maximum size set of in-trees $\mathcal{T}$ rooted at the given root $r \in V$ such that

$$\sum_{T \in \mathcal{T}} c(T, v) \leq b(v), \ \forall v \in V.$$

Note that the set of in-trees $\mathcal{T}$ is a multiset, i.e., it may include same in-trees.

The aim of this paper is to reveal the computational complexity of several variations of the problem. We consider general graphs, acyclic graphs and complete graphs embedded in the d-dimensional space $\mathbb{R}^d$ as inputs. When the graph is embedded in $\mathbb{R}^d$, we assume that the consumptions of each edge depend only on the distance between its end nodes, and we call such a problem *metric*. More precisely, let $(x_1(v), \ldots, x_d(v)) \in \mathbb{R}^d$ be the d-dimensional coordinate of a node v. We denote the L^p distance between v and w as $L^p(v, w) = (|x_1(v) - x_1(w)|^p + \cdots + |x_d(v) - x_d(w)|^p)^{1/p}$ for $1 \leq p < \infty$ and $L^p(v, w) = \max\{|x_1(v) - x_1(w)|, \ldots, |x_d(v) - x_d(w)|\}$ for $p = \infty$. In this paper, the parameter $p \in \{1, \ldots, \infty\}$ is a fixed constant. The head and tail consumption functions are called metric if their values for each edge are proportional to the L^p distance between its end nodes (i.e., there exist constants α and β such that for each edge $e = (v, w)$ in E, $h(e) = \alpha L^p(v, w)$ and $t(e) = \beta L^p(v, w)$ hold), and a problem is metric if the consumption functions are metric. Our results are summarized as follows:

- packing one in-tree
  - without head consumptions: polynomially solvable
  - with identical head consumptions: NP-hard in the strong sense
  - with head consumptions on acyclic graphs: NP-hard in the strong sense
  - with metric head consumptions: NP-hard in the strong sense
- packing in-trees (in general)
  - without head consumptions
    * on acyclic graphs: polynomially solvable
    * on general graphs: NP-hard in the strong sense
    * with metric tail consumptions: NP-hard in the strong sense
  - with head consumptions
    * with identical consumptions on acyclic graphs: polynomially solvable
    * in general: NP-hard in the strong sense

Recently, several kinds of graph packing problems are studied in the context of ad hoc wireless networks and sensor networks. These problems are called *network lifetime problems*. Included among the important problems in this category are the node capacitated spanning subgraph packing problems considered in [1, 9, 12, 15]. For sensor networks, for example, a spanning subgraph corresponds to a communication network topology for collecting information from all nodes (sensors) to the root (base station) or for sending information from the root to all other nodes. Sending a message along an edge consumes energy at end nodes, usually depending on the distance between them. Battery capacities of sensor nodes are severely limited. It is therefore important to design the topologies for communication to save energy consumption and make sensors operate as long as possible. The merit of designing such topologies in a centralized manner was indicated in [9], in which a cluster-based network organization algorithm, called LEACH-C (low energy adaptive clustering hierarchy centralized), was proposed. As the topology for communication, they considered in-trees with a bounded height. For more energy efficient communication networks, a multi-round topology construction problem was formulated as a mathematical programming problem, and a heuristic solution method was proposed [12]. The objective of the problem is to maximize the number of packed in-trees (n.b., spanning in-trees without any additional constraints are considered). Most recently, an effective heuristic algorithm for this problem was proposed [15]. In the formulation of [1], head consumptions are not considered, and the consumption at each node is the maximum tail consumption among the edges leaving the node. There are variations of the problem with respect to additional conditions on the spanning subgraph such as strong connectivity, symmetric connectivity, and being a directed out-tree rooted at a given node. The authors discussed the hardness of the problem and proposed several approximation algorithms.

These network lifetime problems, including our problem, are similar to the well-known *edge-disjoint spanning arborescence packing problem*: Given a directed graph $G = (V, E)$ and a root r in V, find the maximum number of edge-disjoint arborescences rooted at r. Note that the given graph G may have parallel edges. The edge-disjoint arborescence packing problem is solvable in polynomial time [3, 10]. Its capacitated version is also solvable in polynomial time [5, 11, 14].

The rest of this paper is organized as follows. In Section 2, we consider the computational complexity of a special case of our problem, packing one in-tree. We show that the case without head consumptions is tractable and that the case with head consumptions is intractable even when the given graph is acyclic or metric. In Section 3, we consider the complexity of the problem in general. We show that the problem with identical consumptions on acyclic graphs is tractable and that the problem without head consumptions is intractable even when the given graph is metric. In Section 4, we study the fractional packing, that is, the linear relaxation of the problem.

Throughout this paper, we assume that the given root node is reachable from all other nodes, because otherwise the node capacitated in-tree packing problem is meaningless. We also assume that $c(T) > 0$ for any in-tree T, where $c(T) = \sum_{v \in V} c(T, v)$; otherwise we can pack any number of an in-tree T with $c(T) = 0$. The second assumption implies that the objective value k of any solution $\mathcal{T}$ satisfies $k \leq \sum_{v \in V} b(v) / \min\{c(T)\}$, which leads to the property that $\log k$ is bounded by a polynomial of the input size.

2 Packing One In-Tree

In this section, we consider a decision problem, which we call the node capacitated one in-tree packing problem: Given an instance of the node capacitated in-tree packing problem, decide whether the instance has a packing with one in-tree. Below we show that though the problem without head consumptions is tractable, the problem with head consumptions is intractable in some restricted cases.

2.1 Tractable Cases

Here we first consider the problem without head consumptions, i.e., $h(e) = 0$, $\forall e \in E$. In this case, the consumption of a spanning in-tree at each node is caused by exactly one edge leaving the node. Let $E' \subseteq E$ be the set of all edges whose tail consumptions are at most the capacity of their tail nodes, i.e., $\forall e' \in E'$, $t(e') \leq b(\mathrm{tail}(e'))$, where $\mathrm{tail}(e)$ is the tail node of a directed edge e. Because the node capacitated one in-tree packing problem on (V, E) is equivalent to the problem of finding an in-tree in (V, E'), the following lemma holds.

Lemma 1 *The node capacitated one in-tree packing problem without head consumptions is solvable in* $\mathrm{O}(|E|)$ *time.*

2.2 Intractable Cases

Here we consider the problem with head consumptions. In this case, the node capacitated one in-tree packing problem is NP-complete in the strong sense.

Theorem 1 *The node capacitated one in-tree packing problem is NP-complete in the strong sense. The problem is still NP-complete in the strong sense, even when the given graph is acyclic and there are no tail consumptions.*

Proof: We show that the bin-packing problem polynomially transforms to the above decision problem without tail consumptions. The bin-packing is known to be NP-complete in the strong sense [6].

Let $\{1, \ldots, k\}$ be the set of bins, where the capacity of each bin is a positive integer B. Let n be the number of items to be packed into bins and s_i $(i \in \{1, \ldots, n\})$ be the sizes of items satisfying $s_i \in \mathbb{Z}_+$, where $\mathbb{Z}_+$ is the set of nonnegative integers. We note that, as the bin-packing problem is NP-complete in the strong sense, the bin-packing problem is NP-complete even when $\max\{s_i : i \in \{1, \ldots, n\}\}$ and B are bounded by a polynomial of n.

For the set of k bins, we introduce a set of nodes $U := \{u_1, \ldots, u_k\}$. For each item i, we introduce a node w_i. Let $W := \{w_1, \ldots, w_n\}$. We also introduce a root node r. Edges are emanating from each node in W to all nodes in U, and emanating from each node in U to the root r. We set a head consumptions $h(e) := s_i$ if the edge e is leaving $w_i \in W$, and 0 otherwise. We set the capacity of a node to B if the node corresponds to a bin, and 0 otherwise. See Figure 1 for an example. The size of the resulting instance

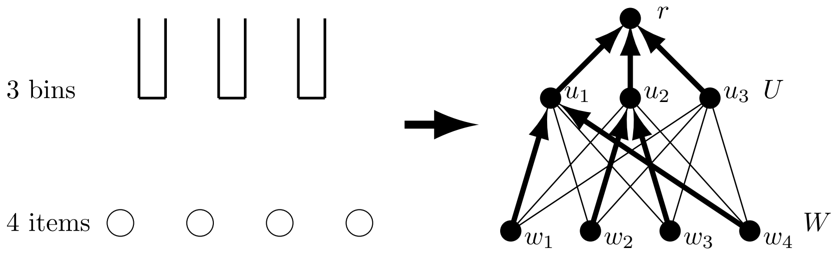

Figure 1: An example of the transformation from the bin-packing problem

of the in-tree packing problem is bounded by a polynomial of k and n, and the largest number in the instance is the same as that of bin-packing problem.

Suppose that the in-tree packing problem has a feasible solution. Because every edge emanating from W enters U in the instance, for each $i \in \{1, \ldots, n\}$ there exists an index $j \in \{1, \ldots, k\}$ such that the in-tree contains (w_i, u_j). It means that there is a corresponding feasible bin-packing such that the item i is packed into the bin j. Similarly, the opposite holds. $\square$

Next we consider the case that the given graph is not necessarily acyclic. The problem is NP-complete, even when the given head consumptions are identical and there are no tail consumptions.

Theorem 2 *The node capacitated one in-tree packing problem is NP-complete in the strong sense, even when the given graph is planar, the given head consumptions are identical and there are no tail consumptions.*

Proof: We show that the Hamiltonian path problem polynomially transforms to the above decision problem without tail consumptions. The Hamiltonian path is known to be NP-complete in the strong sense, even when the given graph is planar and an end point of the path is specified [6].

Let $G = (V, E)$ be the given graph and $r \in V$ be the specified end point of the Hamiltonian path. We introduce the head consumption $h(e) := 1$, $\forall e \in E$. We set the capacity $b(v) := 1$, $\forall v \in V$. Let $r \in V$ be the root node.

Suppose that the in-tree packing problem has a feasible solution. Because all head consumptions and all node capacities are 1, the in-tree must be a directed path that includes all nodes and ends at the root node, i.e., a Hamiltonian path. Similarly, the opposite holds. □

The metric case of the problem is also NP-hard even when the graph is a complete graph embedded in the 1-dimensional space $\mathbb{R}^1$ and there are no tail consumptions.

Theorem 3 *The node capacitated one in-tree packing problem is NP-complete in the strong sense, even when there are no tail consumptions and the given graph is a complete graph embedded in the 1-dimensional space* $\mathbb{R}^1$.

Proof: We show that the bin-packing problem polynomially transforms to the above decision problem such that there are no tail consumptions and the given graph is embedded in $\mathbb{R}^1$.

Let $\{1, \ldots, k\}$ be the set of bins whose capacities are $B \in \mathbb{Z}_+$. Let n be the number of items to be packed into bins and $s_i \in \mathbb{Z}_+$ ($i \in \{1, \ldots, n\}$) be the sizes of items. We assume that $\max\{s_i : i \in \{1, \ldots, n\}\}$ and B are bounded by a polynomial of n; even with this restriction, the bin-packing problem is NP-complete [6].

For the set of k bins, we introduce a set of nodes $U := \{u_1, \ldots, u_k\}$ whose coordinates (i.e., positions in $\mathbb{R}^1$) are 0. For each item i, we introduce a node w_i whose coordinate is ks_i. Let $W := \{w_1, \ldots, w_n\}$. We also introduce a root node r whose coordinate is -1. All pairs of nodes are connected with each other. The head consumption of each edge is the distance between its end nodes. The capacity $b(v)$ of each node v is defined as follows:

$$b(v) := \begin{cases} k & (v = r), \\ kB & (v \in U), \\ 0 & (v \in W). \end{cases}$$

The size of the resulting instance and the largest number in it are bounded by a polynomial of k and n.

Suppose that the metric in-tree packing problem has a feasible solution. Because the capacity of the root node is k and the distance between each node w_i and the root is at least $k+1$, the edge (w_i, r) never appears in the in-tree. As a result, for each $i \in \{1, \ldots, n\}$, there exists an index $j \in \{1, \ldots, k\}$ such that the in-tree contains (w_i, u_j). This means that there is a corresponding feasible bin-packing such that the item i is packed into the bin j. Similarly, the opposite holds. □

3 Packing In-Trees (in General)

In this section, we study the general case of our problem, i.e., the maximization of the number of packed in-trees. Though many restricted cases of our problem are NP-hard, identical consumptions on acyclic graphs make our problem tractable. Before we discuss the intractability of our problem, we show that our problem belongs to the class NP.

3.1 Compact Representation and Class NP

From the definition of our problem, we allow a feasible solution whose size is not bounded by a polynomial order of the input size. Note that, though we defined a solution $\mathcal{T}$ as a multiset, it can be represented compactly by specifying the set of distinct in-trees in $\mathcal{T}$ and the number of appearance of each in-tree. The size of a solution signifies the number of different in-trees and the digits needed to specify their appearance numbers. In this subsection, we show that for any feasible solution to the in-tree packing problem, there exists a solution having the same objective value and it can be represented in a polynomial order of the input size. This implies that the node capacitated in-tree packing problem belongs to the class NP. To show this, we use the fact that the capacitated version of the edge-disjoint arborescence packing problem is solvable in polynomial time [5].

Theorem 4 *Let $\mathcal{T} := \{T_1, \ldots, T_k\}$ be a solution (i.e., it can be a multiset) of the node capacitated in-tree packing problem. Then there is a solution such that the objective value is k and it can be represented in a polynomial order of the input size. More precisely, there exists a solution with objective value k consisting of at most $|V| + |E| - 2$ distinct in-trees.*

Proof: We show a simple way of constructing a polynomial size solution from any solution preserving the objective value. For this, we use a polynomial time algorithm for the (capacitated version of the) edge-disjoint spanning arborescence packing problem, which we call algorithm A. Let $G = (V, E)$ be the given graph of the node capacitated in-tree packing problem. We introduce an edge capacity function $\omega : E \to \mathbb{R}_+$. The capacity $\omega(e)$ of each edge e is defined as the number of in-trees in $\mathcal{T}$ that include e, i.e., $\omega(e) = |\{T_l \in \mathcal{T} : e \in T_l\}|$ (recall that even if two in-trees T_l and $T_{l'}$ are the same, they are considered as different elements in $\mathcal{T}$). The size of the resulting edge-disjoint spanning arborescence packing problem is bounded by a polynomial of the input size of the original instance of the node capacitated in-tree packing problem, because the size of the graph is the same and the values of $\omega(e)$ are bounded by k (recall that $\log k$ is polynomially bounded as mentioned in Section 1). Then we apply algorithm A to solve the edge-disjoint arborescence packing problem on G with edge capacities defined by ω. The objective value of the optimal arborescence packing is k. (Because $\mathcal{T}$ is a feasible solution to the edge disjoint arborescence packing problem, the optimal value of the arborescence packing is at least k. Because the sum of edge capacities for all edges is $k \times (|V| - 1)$, it is impossible to pack $k + 1$ arborescences.) The size needed to describe the obtained optimal solution is bounded by a polynomial of the input size, because the size of the output cannot exceed the time complexity of the algorithm A that obtained it. If an algorithm in [5] is used as algorithm A, an optimal solution for the edge-disjoint arborescence packing problem whose number of distinct arborescences is at most $|V| + |E| - 2$ can be found in polynomial time. The feasible solution for the edge-disjoint arborescence packing problem is a feasible solution for the node capacitated in-tree packing problem because the feasibility depends only on the total number of times each edge appears in the arborescences (or in-trees). $\square$

This theorem directly yields the following corollary.

Corollary 1 *The node capacitated in-tree packing problem belongs to the class NP.*

3.2 Tractable Cases

We consider the case where the given graph is acyclic. When a given graph is acyclic, an in-tree of the graph is easily found.

Proposition 1 *When a graph $G = (V, E)$ is acyclic, a set of edges $E' \subseteq E$ is an in-tree if the outdegree in (V, E') is one for all nodes except for the root.*

From Proposition 1, we can easily see that the node capacitated in-tree packing problem without head consumptions is solvable in $\mathrm{O}(|E|)$ time.

Lemma 2 *When the given graph is acyclic and there are no head consumptions, the node capacitated in-tree packing problem is solvable in $\mathrm{O}(|E|)$ time.*

Proof: Let $G = (V, E)$ be the given graph and $r \in V$ be the given root node. For all $v \in V \setminus \{r\}$, we let $e_{\min}(v)$ be an edge leaving v such that the tail consumption is minimum among all edges leaving v, i.e.,

$$e_{\min}(v) := e \in \delta^+_E(v) \quad \text{such that} \quad t(e) \le t(e'),\ \forall e' \in \delta^+_E(v).$$

Then the set of edges $\{e_{\min}(v) : v \in V \setminus \{r\}\}$ is an in-tree from Proposition 1. An optimal in-tree packing can be obtained by packing this tree as many as possible. $\square$

Though head consumptions make our problem intractable, our problem with identical consumptions remains polynomially solvable.

Theorem 5 *If the given graph $G = (V, E)$ is acyclic and the given consumptions are identical, i.e., $h(e) = t(e) = 1$ for all $e \in E$, then the node capacitated in-tree packing problem is solvable in polynomial time.*

To prove Theorem 5, we show that the following decision problem is solvable in polynomial time: Given an acyclic graph $G = (V, E)$, identical consumptions and a number k, decide whether the instance has a packing of k in-trees. If the decision problem is solvable in polynomial time, the optimization problem is solvable in polynomial time using binary search on the value of k.

To prove that the decision problem is solvable in polynomial time, we use the well-known fact that the maximum flow problem is polynomially solvable. Additionally, it is known that an integral maximum flow can be obtained in polynomial time if the capacity of each edge is integral. Let $F(n, m, C)$ be the time complexity of such an algorithm to solve the maximum flow problem, where n is the number of nodes, m is the number of edges and C is the maximum capacity. For example, $F(n, m, C)$ can be $\mathrm{O}(n^3/\log n)$, $\mathrm{O}(nm\log(n^2/m))$ or $\mathrm{O}(m\min\{n^{2/3}, m^{1/2}\}\log(n^2/m)\log C)$ [14].

Lemma 3 *If the given graph $G = (V, E)$ is acyclic and the given consumptions are identical; $h(e) = t(e) = 1$ for all $e \in E$, then the problem of determining whether there exists a feasible packing of k in-trees is solvable in $\mathrm{O}(|V||E| + F(|V|, |E|, \max\{k, \max_v b(v) - k\}))$ time.*

Proof: Let $V = \{v_0, v_1, \ldots, v_n\}$ be the set of nodes where v_0 is the root node. If there is a node $v_j \in V \setminus \{v_0\}$ whose capacity $b(v_j)$ is less than k, then there is no feasible packing of k in-trees. In this proof, we assume without loss of generality that $b(v_j)$ is an integer for all $v_j \in V$, because if node capacity $b(v_j)$ is fractional for some node v_j, we can replace it with $\lfloor b(v_j) \rfloor$. We show that the problem is polynomially reduced to the maximum flow problem.

For the set of nodes $V = \{v_0, v_1, \ldots, v_n\}$, we introduce two sets of nodes $U := \{u_1, u_2, \ldots, u_n\}$ and $W := \{w_0, w_1, \ldots, w_n\}$. We also introduce a source node s and a sink node z. For the set of edges E, we introduce a set of edges $E' := \{(u_i, w_j) : (v_i, v_j) \in E\}$. Furthermore, edges are emanating from the source node s to each node in U and emanating from each node in W to the sink node z. See Figure 2 for an example. The capacity of each edge from w_j ($j \geq 1$) to z is $b(v_j) - k$, and those of the other edges (i.e., edges from s to U, U to W, and the edge from w_0 to z) are set to k. This reduction can be done in $\mathrm{O}(|V| + |E|)$ time.

Because the resulting network has $\mathrm{O}(|V|)$ nodes and $\mathrm{O}(|V| + |E|) = \mathrm{O}(|E|)$ edges, and the capacity of each edge is integral, an integral maximum flow from s to z is obtained in polynomial time. Let $q : E' \to \mathbb{Z}_+$ be the obtained integral flow on E'. If the value $\sum_{e \in E'} q(e)$ of the maximum flow is (strictly) less than kn, there is no feasible packing of k in-trees. If the maximum value is kn, we can obtain a feasible packing of k in-trees from q as follows:

(Step 1) For each $i \in \{1, \ldots, n\}$, choose an edge e_i satisfying $q(e_i) > 0$ from those emanating from u_i.

(Step 2) Let $T := \{e_1, \ldots, e_n\}$. Then T is an in-tree because G is acyclic (and from Proposition 1).

(Step 3) Let $q_{\min} := \min_i\{q(e_i)\}$. Output the in-tree T to be packed $q_{\min}$ times.

(Step 4) For each $e_i \in T$, decrease the flow value $q(e_i)$ by $q_{\min}$.

(Step 5) If there is an edge $e \in E'$ such that $q(e) > 0$, then go to Step 1; otherwise stop.

Each step of the above procedure is executed in $\mathrm{O}(|V|)$ time and the number of iterations is at most $|E|$, because in each iteration the flow value of at least one edge emanating from U becomes 0. Hence the time complexity of the entire algorithm is $\mathrm{O}(|V||E|)$ plus the time to solve the maximum flow problem on a graph with $\mathrm{O}(|V|)$ nodes, $\mathrm{O}(|E|)$ edges and maximum edge capacity $\max\{k, \max_v b(v) - k\}$. □

It is easy to extend Theorem 5 to the following theorem.

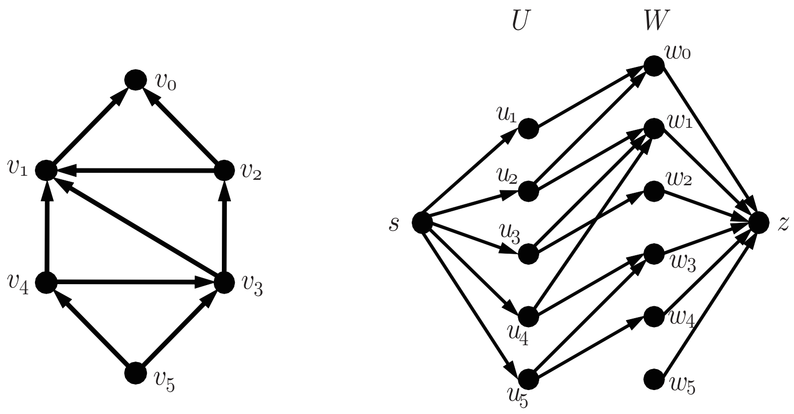

Figure 2: An example of the reduction to the maximum flow problem

Theorem 6 *If the given graph is acyclic and the given consumption functions h and t satisfy*

$$h(e) = h(e') \text{ if } e \text{ and } e' \text{ share a common head, and}$$
$$t(e) = t(e') \text{ if } e \text{ and } e' \text{ share a common tail,}$$

then the node capacitated in-tree packing problem is solvable in polynomial time.

Proof: For the simplicity of notation, let $h(v)$ denote the head consumption of each edge entering v, and let $t(v)$ denote the tail consumption of each edge emanating from v. The theorem can be proved by modifying the proof of Lemma 3 as follows. We first check whether $b(v_j)$ is less than $k \cdot t(v_j)$ for each node $v_j \in V \setminus \{v_0\}$. If such a node exists, it is impossible to pack k in-trees. Otherwise, we use a similar reduction to the maximum flow problem in which we replace the capacity $b(v_j) - k$ with $\lfloor (b(v_j) - k \cdot t(v_j))/h(v_j) \rfloor$ for each edge from w_j $(j \geq 1)$ to z. □

3.3 Intractable Cases

Though having no head consumptions makes our problem tractable in the case of packing one in-tree, our problem without head consumptions is intractable in the case of packing in-trees in general.

Theorem 7 *The following problem is NP-complete in the strong sense: Given an instance of the node capacitated in-tree packing problem and a number n, decide whether the instance has a packing of n in-trees. The problem is still NP-complete in the strong sense, even when there are no head consumptions.*

Proof: We show that the bin-packing problem (known to be NP-complete in the strong sense) polynomially transforms to the above decision problem without head consumptions.

Let $\{1, \ldots, k\}$ be the set of bins whose capacities are $B \in \mathbb{Z}_+$. Let n be the number of items to be packed into bins and s_i $(i = 1, \ldots, n)$ be the sizes of items satisfying $s_i \in \mathbb{Z}_+$.

For the set of k bins, we introduce a set of nodes $U := \{u_1, \ldots, u_k\}$. For an item i, we introduce a node w_i. Let $W := \{w_1, \ldots, w_n\}$. We also introduce a root node r. Edges are emanating from each node in W to the root r, connecting nodes in U and W with each other, and connecting all nodes in U with each other. We set a tail consumption s_i to edges entering w_i, 1 to edges entering the root, and 0 to others. We set the capacity of each node in W to 1, each node in U to B, and the root to 0. See Figure 3 for an example. The size of the resulting instance is bounded by a polynomial of k and n, and the largest number in it is the same as that of bin-packing.

Suppose the node capacitated in-tree packing has a solution $\mathcal{T} := \{T_1, \ldots, T_n\}$. Because the capacity of each node in W is the same as the tail consumption of each edge entering the root, each edge (w_i, r)

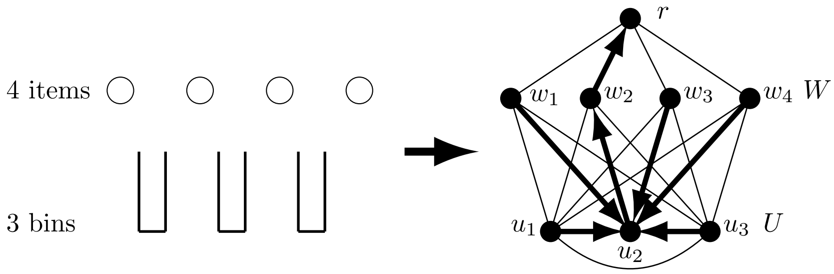

Figure 3: An example of the transformation from the bin-packing problem

appears in exactly one tree in $\mathcal{T}$. Let T_i be the in-tree containing an edge (w_i, r). In the tree T_i, every edge leaving from $W \setminus \{w_i\}$ must enter a node in U. Because T_i is an in-tree, there must be at least one edge leaving U and entering w_i. Let (u_j, w_i) be such an edge. Then, a feasible solution to the bin-packing problem is obtained by packing each item i into the bin j. Similarly, the opposite holds. $\square$

In the following, we show that the node capacitated in-tree packing problem is still hard when the problem is *metric*. The metric problem is quite natural in the context of wireless radio networks, because the radio intensity decreases according to the distance [9]. In other words, as the distance between nodes becomes larger, a radio transmitter consumes more energy.

Theorem 8 *The following problem is NP-complete in the strong sense: Given a number n and an instance of the node capacitated metric in-tree packing problem, decide whether the instance has an in-tree packing of objective value n. The problem is still NP-complete in the strong sense, even when there are no head consumptions and the given graph is a complete graph embedded in the 1-dimensional space $\mathbb{R}^1$.*

Proof: We show that the bin-packing problem polynomially transforms to the above decision problem such that there are no head consumptions and the given graph is embedded in $\mathbb{R}^1$.

Let $\{1, \dots, k\}$ be the set of bins whose capacities are $B \in \mathbb{Z}_+$. Let n be the number of items to be packed into bins and $s_i \in \mathbb{Z}_+$ ($i \in \{1, \dots, n\}$) be the sizes of items. We assume that $\max\{s_i : i \in \{1, \dots, n\}\}$ and B are bounded by a polynomial of n; even with this restriction, the bin-packing problem is NP-complete [6].

We define $s_0 = 0$. For simplicity, we first assume that all the items have different sizes, and $0 = s_0 < s_1 < s_2 < \cdots < s_n \leq B$ holds. For the set of k bins, we introduce a set of nodes $U := \{u_1, \dots, u_k\}$ whose coordinates are 0. For each item i, we introduce a node w_i whose coordinate is $3s_i$ and a set of nodes $V_i := \{v_{i1}, v_{i2}, \dots, v_{iN_i}\}$ whose coordinates are $3s_{i-1}+1, 3s_{i-1}+2, \dots, 3s_i - 2$ where $N_i := 3(s_i - s_{i-1}) - 2$. Let $W := \{w_1, \dots, w_n\}$ and $V := V_1 \cup V_2 \cup \cdots \cup V_n$. We also introduce a root node r whose coordinate M is sufficiently large (e.g., $M = 3B + 2n$). All pairs of nodes are connected with each other. The tail consumption of each edge is the distance between its end nodes. The capacity $b(v)$ of each node v is defined as follows:

$$b(v) := \begin{cases} 0 & (v = r), \\ 3B & (v \in U), \\ M - 3s_i + 2(n-1) & (v = w_i \in W), \\ n & (v \in V). \end{cases}$$

See Figure 4 for an example (4 items and 3 bins). Because we assumed that s_i ($\forall i \in \{1, \dots, n\}$) and B are bounded by a polynomial of n, the size of the resulting instance and the largest number in it are bounded by a polynomial of k and n.

Suppose the node capacitated metric in-tree packing has a solution $\mathcal{T} := \{T_1, \dots, T_n\}$. Each node $v \in V$ must connect to the nearest node on its left side for all n in-trees. Because M is sufficiently large, edges connecting nodes in U and r never appear in the in-trees. Because $b(w_i) = M - 3s_i + 2(n-1)$ ($w_i \in W$) and M is sufficiently large, for any index i, the edge (w_i, r) appears at most once in the in-trees. From the setting of $b(w_i)$, once the edge (w_i, r) is used, (w_i, v_{iN_i}) must be used in the rest of in-trees.

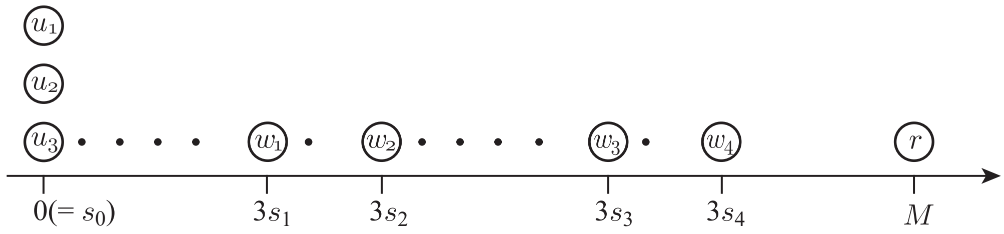

Figure 4: An example of embedded nodes

As a result, for each $i \in \{1, \ldots, n\}$, there exists an index $j \in \{1, \ldots, k\}$ such that the in-tree containing (w_i, r) is composed of (u_j, w_i), (w_i, r), (w_l, v_{lN_l}) for each $w_l \in W \setminus \{w_i\}$, (u_l, u_j) for each $u_l \in U \setminus \{u_j\}$, and edges leaving from each $v \in V$ to its left neighbor. See Figure 5 for an example. Then edges (u_j, w_i) indicate a feasible solution of the bin-packing problem. Similarly, the opposite holds.

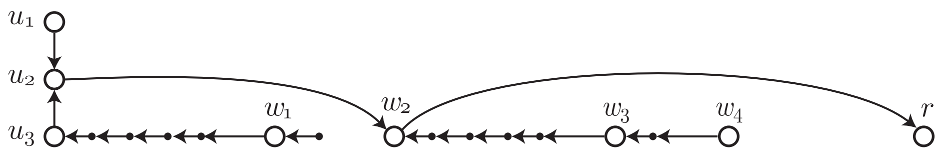

Figure 5: An in-tree of a feasible packing

Using symbolic perturbation technique with additional $\lceil \log_2 n \rceil$ bits [2], a problem instance including items of the same size transforms to the problem such that all the items have different sizes. When the graph is embedded in $\mathbb{R}^d$ and the tail consumption is the L^p distance, the above 1-dimensional case is included. $\square$

4 Fractional Packing

In the node capacitated in-tree packing problem, we maximize the number of in-trees packed. In this section, we consider a fractional packing problem, the linear relaxation problem of our problem.

The *fractional node capacitated in-tree packing problem* is to find a set of in-trees $\mathcal{T}$ rooted at the given root $r \in V$ and a nonnegative weight function $x : \mathcal{T} \to \mathbb{R}_+$ such that

$$\sum_{T \in \mathcal{T}} x(T)c(T, v) \leq b(v), \ \forall v \in V.$$

The objective of the problem is to maximize the sum of weights $\sum_{T \in \mathcal{T}} x(T)$. If the value of x is restricted to integers, this problem is the same as the original node capacitated in-tree packing problem.

Though the node capacitated in-tree packing problem is NP-hard in the strong sense, the fractional node capacitated in-tree packing problem is solvable in polynomial time as explained below. Let $\mathcal{T}_{\text{all}}$ be the set of all in-trees of the given graph. The problem is formulated as following linear programming problem (LPP):

$$
\begin{aligned}
\text{maximize} \quad & \sum_{T \in \mathcal{T}_{\text{all}}} x(T) \\
\text{subject to} \quad & \sum_{T \in \mathcal{T}_{\text{all}}} x(T)c(T, v) \leq b(v), \ \forall v \in V \\
& x(T) \geq 0, \ \forall T \in \mathcal{T}_{\text{all}}.
\end{aligned}
$$

The dual of the above linear programming problem (LPD) is the following:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{v \in V} b(v)y(v) \\
\text{subject to} \quad & \sum_{v \in V} c(T, v)y(v) \geq 1, \ \forall T \in \mathcal{T}_{\text{all}} && (1) \\
& y(v) \geq 0, \ \forall v \in V. && (2)
\end{aligned}
$$

Though the problem (LPD) has an exponential number of constraints, we show that it can be solved in polynomial time.

In the research of the linear programming problem, the *separation problem* is a problem of finding a hyperplane that separates a given solution from the feasible region (e.g., a violated constraint) if it exists. It is well known that, if the separation problem can be solved in polynomial time, the linear programming problem can also be solved in polynomial time [7, 8, 13]. To be more precise, the linear programming problem can be solved and an optimal dual solution can also be found by algorithms that are constructed by using a separation algorithm repeatedly within the ellipsoid method, and the number of calls to the separation algorithm and the number of elementary arithmetic operations executed by each of them are bounded by a polynomial (see Corollaries 14.1a and 14.1g in [13]). Accordingly, if we can solve the separation problem for the problem (LPD) in polynomial time, the problem (LPP) can be solved in polynomial time.

The separation problem for the problem (LPD) is to find a violated constraint among constraints of (1)–(2) for a given y if it exists. It is trivial that the validity of constraints of (2) can be ensured in polynomial time. However, for constraints of (1), there are an exponential number of constraints and it is unable to check them one by one in order to attain a polynomial time algorithm. We transform the left-hand side of (1) as follows:

$$
\begin{aligned}
\sum_{v \in V} c(T, v)y(v) &= \sum_{v \in V} \left(\sum_{e \in \delta_T^-(v)} h(e) + \sum_{e \in \delta_T^+(v)} t(e) \right) y(v) \\
&= \sum_{v \in V} \sum_{e \in \delta_T^-(v)} h(e)y(\text{head}(e)) + \sum_{v \in V} \sum_{e \in \delta_T^+(v)} t(e)y(\text{tail}(e)) \\
&= \sum_{e \in T} h(e)y(\text{head}(e)) + \sum_{e \in T} t(e)y(\text{tail}(e)) \\
&= \sum_{e \in T} \big(h(e)y(\text{head}(e)) + t(e)y(\text{tail}(e)) \big),
\end{aligned}
$$

where head(e) and tail(e) are the head and the tail nodes of a directed edge e. Then, the constraint signifies that when each edge e has a weight $h(e)y(\text{head}(e)) + t(e)y(\text{tail}(e))$, the total weight of every in-tree T must be at least 1. Hence, for a given y, if the minimum weight of in-trees is greater than or equal to 1, we can conclude that y satisfies condition (1) for all $T \in \mathcal{T}_{\text{all}}$. Otherwise, the constraint corresponding to the minimum weight in-tree is a violated constraint. Note that the minimum weight in-tree problem can be solved in $\mathrm{O}(|V| \log |V| + |E|)$ time [4].

In summary, we have the following theorem.

Theorem 9 *The fractional node capacitated in-tree packing problem can be solved in polynomial time.*

5 Conclusion

In this paper, we considered a node capacitated in-tree packing problem, which has applications in wireless ad hoc networks and sensor networks.

We revealed the computational complexity of the problem under various restrictions on consumption functions and graphs. For the problem of packing one in-tree, we showed that the problem without head consumptions is polynomially solvable and that the problem with identical head consumptions, the problem with head consumptions on acyclic graphs, and the problem with metric head consumptions are NP-hard in the strong sense. For the problem of packing in-trees in general, we proved that the problem without head consumptions and the problem with metric tail consumptions are both NP-hard in the strong sense. We also showed that the problem on acyclic graphs are polynomially solvable if head and tail consumptions are identical or if there are no head consumptions.

One of the most interesting findings in this paper is that the problem remains NP-hard in the strong sense even when the given graph is embedded in the 1-dimensional space and tail consumptions are metric. The metric case often appears in practice, and our result indicates that even very restricted versions of such practical instances are difficult.

Most of the results in this paper were negative, showing that restricted special cases of the node capacitated in-tree packing problem are NP-hard. It will be valuable to investigate well-solvable cases that are practically important.

Acknowledgement

We thank Naoyuki Kamiyama for his valuable comments on the edge-disjoint arborescence packing problem.

References

[1] G. Calinescu, S. Kapoor, A. Olshevsky, and A. Zelikovsky. Network lifetime and power assignment in ad hoc wireless networks. In *Proceedings of the 11th European Symposium on Algorithms*, Vol. 2832 of *Lecture Notes in Computer Science*, pp. 114–126. Springer, 2003.

[2] H. Edelsbrunner and E. P. Mücke. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. *ACM Transactions on Graphics*, 9:66–104, 1990.

[3] J. Edmonds. Edge-disjoint branchings. In *Combinatorial Algorithms*, pp. 91–96. Algorithmic Press, 1973.

[4] H. N. Gabow, Z. Galil, T. Spencer, and R. E. Tarjan. Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. *Combinatorica*, 6(2):109–122, 1986.

[5] H. N. Gabow and K. S. Manu. Packing algorithms for arborescences (and spanning trees) in capacitated graphs. *Mathematical Programming*, 82:83–109, 1998.

[6] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[7] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[8] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 2nd edition, 1993.

[9] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1:660–670, 2002.

[10] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer-Verlag, 4th edition, 2007.

[11] W. Mader. On n-edge-connected digraphs. *Annals of Discrete Mathematics*, 17:439–441, 1983.

[12] M. Sasaki, T. Furuta, F. Ishizaki, and A. Suzuki. Multi-round topology construction in wireless sensor networks. In *Proceedings of the Asia-Pacific Symposium on Queueing Theory and Network Applications*, pp. 377–384, 2007.

[13] A. Schrijver. *Theory of linear and integer programming*. John Wiley and Sons, 1986.

[14] A. Schrijver. *Combinatorial Optimization*. Springer-Verlag, 2003.

[15] Y. Tanaka, M. Sasaki, and M. Yagiura. A heuristic algorithm for the node capacitated in-tree packing problem. In *Proceedings of International Symposium on Scheduling*, pp. 58–64, 2009.