RAIRO-Oper. Res. 55 (2021) 2129–2140
https://doi.org/10.1051/ro/2021096

RAIRO Operations Research
www.rairo-ro.org

ON STAR FAMILY PACKING OF GRAPHS

Mengya Li and Wensong Lin*

Abstract. Let $\mathcal{H}$ be a family of graphs. An $\mathcal{H}$-packing of a graph G is a set $\{G_1, G_2, \ldots, G_k\}$ of disjoint subgraphs of G such that each G_j is isomorphic to some element of $\mathcal{H}$. An $\mathcal{H}$-packing of a graph G that covers the maximum number of vertices of G is called a maximum $\mathcal{H}$-packing of G. The $\mathcal{H}$-packing problem seeks to find a maximum $\mathcal{H}$-packing of a graph. Let i be a positive integer. An i-star is a complete bipartite graph $K_{1,i}$. This paper investigates the $\mathcal{H}$-packing problem with $\mathcal{H}$ being a family of stars. For an arbitrary family $\mathcal{S}$ of stars, we design a linear-time algorithm for the $\mathcal{S}$-packing problem in trees. Let t be a positive integer. An $\mathcal{H}$-packing is called a t^+-star packing if $\mathcal{H}$ consists of i-stars with $i \geq t$. We show that the t^+-star packing problem for $t \geq 2$ is NP-hard in bipartite graphs. As a consequence, the 2^+-star packing problem is NP-hard even in bipartite graphs with maximum degree at most 4. Let T and t be two positive integers with $T > t$. An $\mathcal{H}$-packing is called a $T \setminus t$-star packing if $\mathcal{H} = \{K_{1,1}, K_{1,2}, \ldots, K_{1,T}\} \setminus \{K_{1,t}\}$. For $t \geq 2$, we present a $\frac{t}{t+1}$-approximation algorithm for the $T \setminus t$-star packing problem that runs in $\mathcal{O}(mn^{1/2})$ time, where n is the number of vertices and m the number of edges of the input graph. We also design a $\frac{1}{2}$-approximation algorithm for the 2^+-star packing problem that runs in $\mathcal{O}(m)$ time, where m is the number of edges of the input graph. As a consequence, every connected graph with at least 3 vertices has a 2^+-star packing that covers at least half of its vertices.

Mathematics Subject Classification. 05C70, 68R10.

Received March 16, 2021. Accepted June 23, 2021.

1. Introduction

Let $\mathcal{H}$ be a set of graphs. An *$\mathcal{H}$-packing* Γ of G is a collection of vertex-disjoint subgraphs of G in which each subgraph is isomorphic to some element of $\mathcal{H}$. The vertices in Γ are said to be *covered* and the other vertices of G are said to be *exposed*. The *cardinality* of Γ is the number of vertices covered by Γ. An $\mathcal{H}$-packing of maximum cardinality is called a *maximum $\mathcal{H}$-packing* of a graph. The $\mathcal{H}$-packing problem seeks to find a maximum $\mathcal{H}$-packing of a graph. An *$\mathcal{H}$-factor* (or perfect $\mathcal{H}$-packing) of a graph is an $\mathcal{H}$-packing that covers all vertices of the graph. The $\mathcal{H}$-factor problem is to determine whether a graph has an $\mathcal{H}$-factor.

Let i be a positive integer. We denote by P_i (resp. C_i, K_i) the path (resp. cycle, complete graph) on i vertices. Denote by $K_{1,i}$ the complete bipartite graph with 1 vertex in one part and i vertices in the other. A complete bipartite graph $K_{1,i}$ is called an *i-star*, denoted by S_i. For the case $i \geq 2$, the vertex of degree i in S_i is called the *center* of S_i and a vertex of degree 1 in S_i is called a *leaf* of S_i.

Keywords. Star, tree, $\mathcal{H}$-packing, star family packing, t^+-star packing, $T \setminus t$-star packing.

School of Mathematics, Southeast University, Nanjing 210096, P.R. China.
*Corresponding author: wslin@seu.edu.cn

Let $\mathcal{H} = \{K_2, F_1, \ldots, F_k\}$. It was proved in [7] that if each F_i is hypomatchable then there is a polynomial-time algorithm to find a maximum $\mathcal{H}$-packing of any input graph. Let $\mathcal{H}$ be any subset of $\{K_t \mid t \geq 1\}$. It was mentioned in [7] that if $K_1 \in \mathcal{H}$ or $K_2 \in \mathcal{H}$, then the $\mathcal{H}$-packing problem can be solved in polynomial time, otherwise it is NP-complete. A complete classification of the complexities of $\mathcal{H}$-packing problems for $\mathcal{H} = \{K_2, F\}$ was given in [11]. Let $\mathcal{H}$ be a set of cycles. Then the $\mathcal{H}$-packing problem is NP-complete unless $\mathcal{H}$ is composed of all cycles, all cycles except C_3, or all cycles except C_4 (see [1,9]). A *k-piece* is a connected graph with maximum degree equal to k. If $\mathcal{H}$ is the family of all k-pieces, then the $\mathcal{H}$-packing problem can be solved in polynomial time [6].

Hell and Kirkpatrick [8] studied $\mathcal{H}$-packing problems for $\mathcal{H}$ being a set of complete bipartite graphs. In particular, they showed that a graph has a $\{K_{1,1}, K_{1,2}, \ldots\}$-factor if and only if it contains no isolated vertices. A set of stars is called a *sequential star set* if it is of the form $\{K_{1,1}, K_{1,2}, \ldots\}$ or $\{K_{1,1}, K_{1,2}, \ldots, K_{1,k}\}$ for some $k \geq 1$. Let $\mathcal{H}$ be a set of stars. They proved that the $\mathcal{H}$-factor problem is NP-complete unless $\mathcal{H}$ is a sequential star set. For any fixed integer $k \geq 2$, using a method involving augmenting configurations similar to augmenting paths in the maximum matching problem, they designed an algorithm for the $\{K_{1,1}, K_{1,2}, \ldots, K_{1,k}\}$-packing problem running in time $\mathcal{O}(|V| \cdot |E|)$. In 2011, Bahenko and Gusakov [2] reduced the $\{K_{1,1}, K_{1,2}, \ldots, K_{1,k}\}$-packing problem to the maximum flow problem and thus obtained an exact algorithm that runs in time $\mathcal{O}(\sqrt{|V|} \cdot |E|)$. Kelmans [10] investigated the induced star packing problem of graphs. Let G be a graph and f a function from $V(G)$ to the positive integers. An (f)-star packing of G is a subgraph H of G such that each component of H is a star and if some vertex v is the center of the star then $d_H(v) \leq f(v)$, where $d_H(v)$ is the number of edges incident with v in H. Ning [12] investigated the (f)-star packing problem of graphs. There is also some results on oriented star packings of digraphs [3].

The above results motivate us to investigate star family packing of graphs, where the star family is different from $\{K_{1,1}, K_{1,2}, \ldots, K_{1,k}\}$. In Section 2, for an arbitrary family $\mathcal{S}$ of stars, we design a linear-time algorithm for the $\mathcal{S}$-packing problem in trees. Let t be a positive integer. An $\mathcal{H}$-packing is called a t^+-star packing if $\mathcal{H}$ consists of i-stars with $i \geq t$. In Section 3, we show that the t^+-star packing problem for $t \geq 2$ is NP-hard in bipartite graphs. As a consequence, the 2^+-star packing problem is NP-hard even in bipartite graphs with maximum degree at most 4. Let T and t be two positive integers with $T > t$. An $\mathcal{H}$-packing is called a $T \setminus t$-star packing if $\mathcal{H} = \{K_{1,1}, K_{1,2}, \ldots, K_{1,T}\} \setminus \{K_{1,t}\}$. In Section 4, for $t \geq 2$, we present a $\frac{t}{t+1}$-approximation algorithm for the $T \setminus t$-star packing problem that runs in $\mathcal{O}(mn^{1/2})$ time, where n is the number of vertices and m the number of edges of the input graph. We also design a $\frac{1}{2}$-approximation algorithm for the 2^+-star packing problem that runs in $\mathcal{O}(m)$ time, where m is the number of edges of the input graph. As a consequence, every connected graph with at least 3 vertices has a 2^+-star packing that covers at least half of its vertices.

Let G be a graph. For a vertex v of G, denote by $N(v)$ the set of all neighbors of v. If W is a subset of $N(v)$ then we use $S(v; W)$ to denote the star of G with center v and leaf set W. In case $W = N(v)$, $S(v; W)$ is simply written as $S(v)$.

2. A linear-time algorithm for arbitrary star family packing problems in trees

Let I be a set of positive integers. An *I-star packing* of a graph G is an $\mathcal{H}$-packing of G with $\mathcal{H} = \{i\text{-star}: i \in I\}$. In this section, we design a linear-time algorithm for the I-star packing problem in trees for any set I of positive integers.

Let $T(r)$ be a tree rooted at r. The *level* of a vertex v, denote by $l(v)$, is the length of the path rTv. Let u and v be two vertices such that u is on the path rTv. Then we say u is an *ancestor* of v and v is a *descendant* of u. An ancestor or descendant of a vertex is *proper* if it is not the vertex itself. The immediate proper ancestor of a vertex v other than the root is its *parent*, denoted by $F(v)$. If $F(v) \neq r$, then the parent of $F(v)$ is called the *grandparent* of v, denoted by $\mathrm{GF}(v)$. The vertices whose parent is v are its children. Denote by $\mathrm{CHD}(v)$ the set of all children of v. For any two vertices u and v, if $F(u) = F(v)$ then they are *siblings*. We denote by $B[v]$ the set of all siblings of v including itself.

For a vertex v of $T(r)$, by $T(v)$ we denote the subtree of $T(r)$ rooted at v that is induced by all descendants of v. If $v \neq r$, by $T^*(v)$ we denote the subtree of $T(r)$ obtained from $T(v)$ by adding the vertex $F(v)$ and the edge $vF(v)$.

Problem 1. Maximum I-star Packing in Trees.
Input: A rooted tree $T(r)$ and a set I of positive integers.
Output: A maximum I-star packing of $T(r)$.

Denote by f_v^0 the number of vertices covered by a maximum I-star packing of $T(v) - v$, f_v^+ the maximum number of vertices covered by an I-star packing of $T(v)$ with v being a center of some star, f_v^* the maximum number of vertices covered by an I-star packing of $T^*(v)$ with v being a center of some star and $F(v)$ being covered, and f_v^- the maximum number of vertices covered by an I-star packing of $T(v)$ with v being a leaf of some star.

Accordingly, we denote by F_v^0 a maximum I-star packing of $T(v) - v$, F_v^+ an I-star packing of $T(v)$ with v being a center of some star that covers f_v^+ vertices of $T(v)$, F_v^* an I-star packing of $T^*(v)$ with v being a center of some star and $F(v)$ being covered that covers f_v^* vertices of $T^*(v)$, and F_v^- an I-star packing of $T(v)$ with v being a leaf of some star that covers f_v^- vertices of $T(v)$.

Let $g = \min\{i : i \in I\}$ and $h = \max\{i : i \in I\}$. W.l.o.g. we also assume $h \leq \Delta(T)$, where $\Delta(T)$ is the maximum degree of T.

Remark. It is clear that if v is a leaf of $T(r)$, then $f_v^0 = f_v^+ = f_v^* = f_v^- = 0$ and $F_v^0 = F_v^+ = F_v^* = F_v^- = \emptyset$. If $d_{T(v)}(v) < g$, then $T(v)$ has no I-star packing with v being a center of some star and so $f_v^+ = 0$ and $F_v^+ = \emptyset$. Similarly, if $d_{T^*(v)}(v) < g$, then $f_v^* = 0$ and $F_v^* = \emptyset$. If each child of v in $T(v)$ has degree less than g, then $f_v^- = 0$ and $F_v^- = \emptyset$.

For each $v \in V(T(r))$, let

$$f_v = \max\left\{f_v^-, f_v^0, f_v^+\right\} \text{ and}$$
$$F_v = F_v^- \left(\text{resp.} F_v^0 \text{ or } F_v^+\right) \text{ if } f_v = f_v^- \left(\text{resp.} f_v^0 \text{ or } f_v^+\right).$$

Then it is clear that $F(v)$ is a maximum I-star packing of $T(v)$. In particular, $F(r)$ is a maximum I-star packing of $T(r)$.

For a positive integer k, denote by $[k]$ the set of integers $1, 2, \ldots, k$. In our algorithm, for a vertex v, ΔF_v^+, ΔF_v^* and ΔF_v^- are three sets of stars. If $f_v^+ = 0$ (resp. $f_v^* = 0$ and $f_v^- = 0$), then $\Delta F_v^+ = \emptyset$ (resp. $\Delta F_v^* = \emptyset$ and $\Delta F_v^- = \emptyset$). If $f_v^+ \neq 0$ (resp. $f_v^* \neq 0$ and $f_v^- \neq 0$), then ΔF_v^+ (resp. ΔF_v^* and ΔF_v^-) contains the unique star in ΔF_v^+ (resp. ΔF_v^* and ΔF_v^-) that covers v.

Algorithm 1. An algorithm for I-star Packing in Trees.

```
Input: a rooted tree $T(r)$ and a set $I$ of positive integers
Output: a maximum $I$-star packing of $T(r)$
1: set $g := \min\{i : i \in I\}$ and $h := \max\{i : i \in I\}$
2: set all vertices uncolored
3: while there is an uncolored leaf $l$ of $T(r)$ do
4:     set $f_l^0 := 0$, $f_l^+ := 0$, $f_l^* := 0$, $f_l^- := 0$, $f_l := 0$
5:         $\Delta F_l^+ := \emptyset$, $\Delta F_l^* := \emptyset$, $\Delta F_l^- := \emptyset$
6:     color $l$ red
7: end while
8: while there is an uncolored vertex $v$ with $\mathrm{CHD}(v) \neq \emptyset$ and all its children being red do
9:     let all children of $v$ be $w_1, w_2, \ldots, w_k$
```

```
10:     set $J := \{i \mid f_{w_i} = f^0_{w_i}, i \in [k]\}$
11:     for $i := 1$ to $k$ do
12:         set $\delta_i := f_{w_i} - f^0_{w_i}$
13:         call Algorithm 2 to sort $\delta_1, \delta_2, \ldots, \delta_k$ (so we assume $\delta_1 \le \delta_2 \le \cdots \le \delta_k$ hereafter)
14:     end for
15:     set $f^0_v := \sum_{i=1}^{k} f_{w_i}$
16:     Call Subroutine 1.1 to compute $f^+_v$ and $\Delta F^+_v$
17:     Call Subroutine 1.2 to compute $f^*_v$ and $\Delta F^*_v$
18:     Call Subroutine 1.3 to compute $f^-_v$ and $\Delta F^-_v$
19:     set $f_v := \max\{f^0_v, f^+_v, f^-_v\}$, and color $v$ red
20: end while
21: Call Subroutine 1.4 to retrieve the maximum $I$-star packing of $T(r)$
```

Subroutine 1.1 (compute f^+_v and ΔF^+_v)

```
1: if $k < g$ then
2:     set $f^+_v := 0$ and $\Delta F^+_v := \emptyset$
3: else if $|J| \in I$ then
4:     set $f^+_v := \sum_{i=1}^{k} f_{w_i} + |J| + 1$ and $\Delta F^+_v := \{S(v; \{w_i | i \in J\})\}$
5: else if $|J| > h$ then
6:     set $f^+_v := \sum_{i=1}^{h} f^0_{w_i} + \sum_{i=h+1}^{k} f_{w_i} + h + 1$ and $\Delta F^+_v := \{S(v; \{w_1, w_2, \ldots, w_h\})\}$
7: else if $|J| < g$ then
8:     set $f^+_v := \sum_{i=1}^{g} f^0_{w_i} + \sum_{i=g+1}^{k} f_{w_i} + g + 1$ and $F^+_v := \{S(v; \{w_1, w_2, \ldots, w_g\})\}$
9: else
10:     set $g_1 := \max\{j \mid j < |J|, j \in I\}$ and $h_1 := \min\{j \mid j > |J|, j \in I\}$
11:     set $m_1 := \sum_{i=1}^{k} f_{w_i} + g_1 + 1$ and $n_1 := \sum_{i=1}^{h_1} f^0_{w_i} + \sum_{i=h_1+1}^{k} f_{w_i} + h_1 + 1$
12:     set $f^+_v := \max\{m_1, n_1\}$
13:     if $f^+_v := m_1$ then
14:         set $\Delta F^+_v := \{S(v; \{w_1, w_2, \ldots, w_{g_1}\})\}$
15:     else
16:         set $\Delta F^+_v := \{S(v; \{w_1, w_2, \ldots, w_{h_1})\}\}$
17:     end if
18: end if
```

Subroutine 1.2 (compute f^*_v and ΔF^*_v)

```
1: if $v \neq r$ then
2:     if $k + 1 < g$ then
3:         set $f^*_v := 0$ and $\Delta F^*_v := \emptyset$
4:     else if $|J| + 1 \in I$ then
5:         set $f^*_v := \sum_{i=1}^{k} f_{w_i} + |J| + 2$ and $\Delta F^*_v := \{S(v; \{F(v)\} \cup \{w_i | i \in J\})\}$
```

```
6:      else if $|J|+1 > h$ then
7:          set $f_v^* := \sum_{i=1}^{h-1} f_{w_i}^0 + \sum_{i=h}^{k} f_{w_i} + h + 1$ and $\Delta F_v^* := \{S(v; \{F(v), w_1, w_2, \ldots, w_{h-1}\})\}$
8:      else if $|J|+1 < g$ then
9:          set $f_v^* := \sum_{i=1}^{g-1} f_{w_i}^0 + \sum_{i=g}^{k} f_{w_i}^0 + g + 1$ and $\Delta F_v^* := \{S(v; \{F(v), w_1, w_2, \ldots, w_{g-1}\})\}$
10:     else
11:         set $g_2 := \max\{j \mid j < |J|+1, j \in I\}$ and $h_2 := \min\{j \mid j > |J|+1, j \in I\}$
12:         set $m_2 := \sum_{i=1}^{k} f_{w_i} + g_2 + 1$ and $n_2 := \sum_{i=1}^{h_2-1} f_{w_i}^0 + \sum_{i=h_2}^{k} f_{w_i} + h_2 + 1$
13:         set $f_v^* := \max\{m_2, n_2\}$
14:         if $f_v^* := m_2$ then
15:             set $\Delta F_v^* := \{S(v; \{F(v), w_1, w_2, \ldots, w_{g_2-1}\})\}$
16:         else
17:             set $\Delta F_v^* := \{S(v; \{F(v), w_1, w_2, \ldots, w_{h_2-1}\})\}$
18:         end if
19:     end if
20: else
21:     set $f_v^* := 0$ and $\Delta F_v^* := \emptyset$
22: end if
```

Subroutine 1.3 (compute f_v^- and ΔF_v^-)

```
1: if $d_{T(v)}(w_i) < g$ for all $i = 1, 2, \ldots, k$ then
2:     set $f_v^- := 0$ and $\Delta F_v^- := \emptyset$
3: else
4:     set $J_1 := \{i \mid d_{T(v)}(w_i) \geq g, i \in [k]\}$
5:     set $f_v^- := \max_{i \in J_1} \left\{ f_{w_i}^* + \sum_{j \in [k] \setminus \{i\}} f_{w_j} \right\}$
6:     set $s := \arg\max_{i \in J_1} \left\{ f_{w_i}^* + \sum_{j \in [k] \setminus \{i\}} f_{w_j} \right\}$
7:     set $\Delta F_v^- := \Delta F_{w_s}^*$
8: end if
```

Subroutine 1.4 (retrieve the maximum I-star packing of $T(r)$)

```
1: set $F_r := \emptyset$
2: label $r$ with $\sharp$
3: while there is a vertex $v$ labeled with $\sharp$, $\mathrm{CHD}(v) \neq \emptyset$ and all its children are unlabeled do
4:     if $f_v = f_v^0$ then
5:         label each vertex in $\mathrm{CHD}(V)$ with $\sharp$
6:     else if $f_v = f_v^+$ then
7:         set $F_r := F_r \cup \Delta F_v^+$
8:         let $S$ be the star in $\Delta F_v^+$
9:         label each vertex in $\mathrm{CHD}(v) \setminus V(S)$ with $\sharp$
```

```
10:         **for** each vertex $u$ in CHD$(v) \cap V(S)$ **do**
11:             label each vertex in CHD$(u)$ with $\sharp$
12:         **end for**
13:     **else**
14:         set $F_r := F_r \cup \Delta F_v^-$
15:         let $S$ be the star in $\Delta F_v^-$
16:         let $u \in$ CHD$(v)$ be the center of $S$
17:         label each vertex in CHD$(v) \setminus \{u\}$ with $\sharp$
18:         **for** each vertex $w$ in CHD$(u)$ **do**
19:             label each vertex in CHD$(w)$ with $\sharp$
20:         **end for**
21:     **end if**
22: **end while**
23: return $f_r$ and $F_r$
```

In Algorithm 1 line 13, it calls Algorithm 2 to sort $\delta_1, \delta_2, \dots, \delta_k$. Because each δ_i is a nonnegative integer that is less than or equal to k, Algorithm 2 can sort them in linear time.

Algorithm 2. Sort Algorithm.

```
Input: $q$ integers $k_1, k_2, \dots, k_q$ with $0 \le k_i \le q$ for $i = 1, 2, \dots, q$
Output: $2q$ nonnegative integers $l_1, l_2, \dots, l_q$, $b_1, b_2, \dots, b_q$ s.t. $l_1 \le l_2 \le \dots \le l_q$ and $l_i = k_{b_i}$ for $i = 1, 2, \dots, q$
1: **for** $j := 0$ to $q$ **do**
2:     set $p_j := 0$
3: **end for**
4: **for** $i := 1$ to $q$ **do**
5:     set $j := k_i$
6:     set $P1[j, p_j] := k_i$ and $P2[j, p_j] := i$
7:     increment $p_j$ by 1
8: **end for**
9: set $t := 1$
10: **for** $j := 0$ to $q$ **do**
11:     **if** $p_j > 0$ **then**
12:         **for** $s := 0$ to $p_j - 1$ **do**
13:             set $l_t := P1[j, s]$ and $b_t := P2[j, s]$
14:             increment $t$ by 1
15:         **end for**
16:     **end if**
17: **end for**
```

Lemma 2.1. *Let $k_1, k_2, \dots, k_q$ be q integers with $0 \le k_i \le q$ for $i = 1, 2, \dots, q$. Then Algorithm 2 correctly sorts $k_1, k_2, \dots, k_q$ in increasing order within time $\mathcal{O}(q)$.*

Lemma 2.2. *Let I be a set of positive integers. Let $T(r)$ be any rooted tree of order n. Then Algorithm 1 will find an I-star packing of $T(r)$ within $\mathcal{O}(n)$ time steps.*

Proof. It obviously takes $\mathcal{O}(n)$ time steps to execute Algorithm 1 from lines 1 to 7.

In each iteration of the second "while" loop, let v be a vertex with children $w_1, w_2, \dots, w_k$. The computations of J and $\delta_1, \delta_2, \dots, \delta_k$ require k comparisons and k substractions. By Lemma 2.1, the sorting of $\delta_1, \delta_2, \dots, \delta_k$ is completed within $\mathcal{O}(k)$ time steps. It is not difficult to check that the number of time steps in computing f_v^0, f_v^+, f_v^* and f_v^- is at most $\mathcal{O}(k)$. Because $\sum_{v \in V(T)} d(v) = 2n - 2$, the total number of time steps in executing the second "while" loop is within $\mathcal{O}(n)$.

The number of time steps needed in executing Subroutine 1.4 consists of the times spending in labeling vertices and constructing the I-star packing F_r, which is clearly at most $\mathcal{O}(n)$.

Thus Algorithm 1 finds an I-star packing of $T(r)$ within $\mathcal{O}(n)$ time steps. $\square$

Theorem 2.3. *Let I be a set of positive integers and let $T(r)$ an input rooted tree. Then Algorithm 1 returns a maximum I-star packing of $T(r)$ within $\mathcal{O}(n)$ time steps, where n is the order of $T(r)$.*

Proof. The time complexity has been determined in Lemma 2.2. We next prove that the output F_r is a maximum I-star packing of $T(r)$. It is clear from Subroutine 1.4 that F_r covers exactly f_r vertices. Thus we only need to show that in each iteration of the second "while" loop, it computes $f_v^0, f_v^+, f_v^*, f_v^-$ and f_v correctly. Let v be a vertex of $T(r)$ with children $w_1, w_2, \ldots, w_k$. Suppose it has computed the correct values of $f_{w_i}^0, f_{w_i}^+, f_{w_i}^*, f_{w_i}^-$ and f_{w_i} for $i = 1, 2, \ldots, k$. Then it is clear from the algorithm that f_v^0 is computed correctly. We next show that so is f_v^+.

Recall that f_v^+ is defined as the maximum number of vertices covered by an I-star packing of $T(v)$ with v being a center of some star. Since v must be a center of some star S, we have to find a set $A \subseteq [k]$ with $|A| \in I$ such that the set of leaves of S is $\{w_i | i \in A\}$. Thus

$$\begin{aligned} f_v^+ &= \max\left\{\sum_{i\in A} f_{w_i}^0 + \sum_{i\in[k]\setminus A} f(w_i) + |A| + 1 :\ A \subseteq [k] \text{ and } |A| \in I\right\} \\ &= \max\left\{\sum_{i=1}^{k} f_{w_i} + 1 + |A| - \sum_{i\in A}\delta_i :\ A \subseteq [k] \text{ and } |A| \in I\right\} \\ &= \sum_{i=1}^{k} f_{w_i} + 1 + \max\left\{|A| - \sum_{i\in A}\delta_i :\ A \subseteq [k] \text{ and } |A| \in I\right\}. \end{aligned}$$

To obtain the correct value of f_v^+, the key is to choose a set $A \subseteq [k]$ with $|A| \in I$ that maximize the value of $|A| - \sum_{i\in A}\delta_i$. If $|J| \in I$, then it is obvious that

$$|J| - \sum_{i\in J}\delta_i = \max\left\{|A| - \sum_{i\in A}\delta_i :\ A \subseteq [k] \text{ and } |A| \in I\right\}.$$

Thus

$$f_v^+ = \sum_{i\in J} f_{w_i}^0 + \sum_{i\in[k]\setminus J} f_{w_i} + |J| + 1 = \sum_{i=1}^{k} f_{w_i} + |J| + 1.$$

Note that $\delta_1, \delta_2, \cdots, \delta_k$ have been sorted so that $\delta_1 \le \delta_2 \le \cdots \le \delta_k$. If $|J| > h$, then the set $A = \{1, 2, \cdots, h\}$ maximizes the value $|A| - \sum_{i\in A}\delta_i$. Therefore $f_v^+ = \sum_{i=1}^{h} f_{w_i}^0 + \sum_{i=h+1}^{k} f_{w_i} + h + 1$. If $|J| < g$, then the set $A = \{1, 2, \cdots, g\}$ maximizes the value $|A| - \sum_{i\in A}\delta_i$, and so $f_v^+ = \sum_{i=1}^{g} f_{w_i}^0 + \sum_{i=g+1}^{k} f_{w_i} + g + 1$. Now suppose $g < |J| < h$ and $|J| \notin I$. Let $A_1 = \{1, 2, \cdots, g_1\}$ and $B_1 = \{1, 2, \cdots, h_1\}$. If $|A| < g$, then $|A| - \sum_{i\in A}\delta_i < |A_1| - \sum_{i\in A_1}\delta_i$, and if $|A| > h$, then it is clear that $|A| - \sum_{i\in A}\delta_i \le |B_1| - \sum_{i\in B_1}\delta_i$. It follows that

$$\max\left\{|A| - \sum_{i\in A}\delta_i :\ A \subseteq [k] \text{ and } |A| \in I\right\} = \max\left\{|A_1| - \sum_{i\in A_1}\delta_i,\ |B_1| - \sum_{i\in B_1}\delta_i\right\}.$$

And f_v^+ is computed correctly.

The proof of f_v^* is similar to that of f_v^+, so we omit it. And it is very clear that the algorithm computes f_v^- and f_v correctly. Thus the theorem follows. $\square$

3. Hardness of the t^+-star packing problem in bipartite graphs

Let $t \geq 2$ be an integer. An $\mathcal{H}$-packing with $\mathcal{H} = \{i\text{-star} \mid i \geq t\}$ of a graph G is called a *t^+-star packing* of G. Let G be a graph. A subset K of $V(G)$ is called a *vertex cover* of G if each edge of G has an end vertex in K. A vertex cover of G with minimum number of vertices is called a *minimum vertex cover* of G.

Problem Vertex Cover
Instance: A graph G and an integer k.
Question: Is there a vertex cover of G of cardinality at most k?

It is well known that the problem Vertex Cover is NP-complete, see [5].

Problem t^+-Star Packing
Instance: A graph G and an integer p.
Quesion: Is there a t^+-star packing of G that covers at least p vertices?

Suppose X and Y are two sets of vertices. We denote by $\overline{E}(X, Y)$ the set of all edges xy with $x \in X$ and $y \in Y$, that is $\overline{E}(X, Y) = \{xy \mid x \in X \text{ and } y \in Y\}$.

Theorem 3.1. *Let $t \geq 2$ be an integer. Then the problem t^+-Star Packing in bipartite graphs is NP-complete.*

Proof. It is clear that the problem is in NP since it can be checked in polynomial time that whether a t^+-star packing covers at least p vertices. We prove the theorem by reducing Vertex Cover to t^+-Star Packing. Let (G, k) be an instance of Vertex Cover, where $V(G) = \{v_1, v_2, \ldots, v_n\}$ and $E(G) = \{e_1, e_2, \ldots, e_m\}$. We construct from (G, k) an instance (H, p) of t^+-Star Packing as follows.

For $i = 1, 2, \ldots, n$, let $U_i = \{u_{i,1}, u_{i,2}, \ldots, u_{i,t-1}\}$, $W_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,t-1}\}$, and $E_i = \{e_j \mid e_j \text{ is incident with } v_i \text{ in } G\}$. Then the graph H is defined as

$$V(H) = V(G) \bigcup E(G) \bigcup \left(\mathop{\cup}_{i=1}^{n} U_i \right) \bigcup \left(\mathop{\cup}_{i=1}^{n} W_i \right),$$
$$E(H) = \bigcup_{i=1}^{n} \left(\overline{E}\left(\{v_i\}, E_i \cup U_i\right) \cup \overline{E}\left(\{u_{i,1}\}, W_i\right) \right).$$

Please see Figure 1 for an illustration. It is clear that H is bipartite and $|V(H)| = m + n(2t - 1)$.

Let $p = m + n(t + 1) - k$. We next show that G has a vertex cover of cardinality at most k if and only if H has a t^+-star packing covering at least p vertices of H.

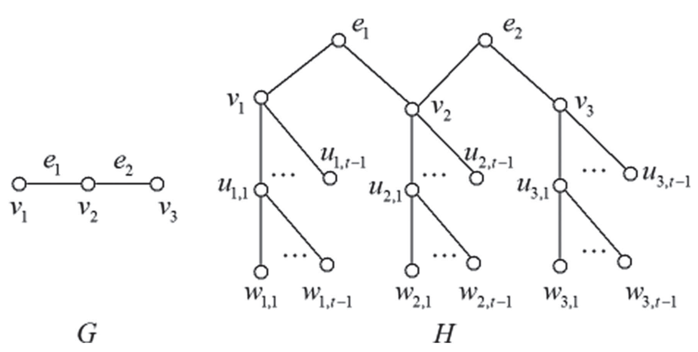

Figure 1. The construction of H.

Suppose G has a vertex cover of cardinality at most k. W.l.o.g., suppose $K = \{v_1, v_2, \ldots, v_q\}$ with $q \le k$ is a minimum vertex cover of G. Because K is a minimum vertex cover, for each $v_i \in K$, there is an edge $e^i \in E_i$ such that $e^i \notin \bigcup_{j\in[q]\setminus\{i\}} E_j$. Set

$$S_i = \begin{cases} S\left(v_i; \{e^i\} \cup U_i\right), & i = 1, 2, \ldots, q \\ S\left(u_{i,1}; \{v_i\} \cup W_i\right), & i = q+1, \cdots, n. \end{cases}$$

Let e be any edge in $E(G) \setminus \{e^1, e^2, \ldots, e^q\}$. Suppose the two end vertices of e are v_i and v_j. Since K is a vertex cover of G, at lease one of v_i and v_j is in K. W.l.o.g., assume $v_i \in K$. Then we add e to the leaf set of the star S_i. After all edges in $E(G) \setminus \{e^1, e^2, \ldots, e^q\}$ have been deposed, we still write the resulting q stars as $S_1, S_2, \ldots, S_q$. Then $S_1, S_2, \ldots, S_q$ cover all vertices of H in $E(G)$. Thus the t^+-star packing $\{S_1, S_2, \ldots, S_n\}$ covers $m + n(t+1) - q \ge m + n(t+1) - k = p$ vertices of H.

Now suppose H has a t^+-star packing covering at least p vertices. We shall prove that G has a vertex cover of cardinality at most k. For a t^+-star packing Γ of H, denote by $C(\Gamma)$ the set of all centers of stars in Γ and by $V(\Gamma)$ the set of vertices covered by Γ. We first prove the following claim.

Claim. Let Γ be a maximum t^+-star packing of H such that $|V(\Gamma) \cap E(G)|$ is as large as possible. Then, (1) $C(\Gamma) \cap E(G) = \emptyset$; (2) $|C(\Gamma) \cap \{v_i, u_{i,1}\}| = 1$ for $i = 1, 2, \ldots, n$; (3) $E(G) \subseteq V(\Gamma)$.

Proof. (1) If $t > 2$ then it is obvious that $C(\Gamma) \cap E(G) = \emptyset$. Thus we assume $t = 2$. Let e_j be any edge in $E(G)$ with two end vertices v_i and v_s. If $e_j \in C(\Gamma)$, then all vertices in $U_i \cup W_i \cup U_s \cup W_s$ are exposed under Γ. Let $\Gamma' = (\Gamma \setminus \{S(e_j; \{v_i, v_s\})\}) \cup \{S(v_i; \{e_j\} \cup U_i), S(u_{s,1}; \{v_s\} \cup W_s)\}$. Then Γ' is a t^+-star packing of H with $|V(\Gamma')| > |V(\Gamma)|$, contradicting the assumption that Γ is a maximum t^+-star packing of H. Therefore $e_j \notin C(\Gamma)$, and so $C(\Gamma) \cap E(G) = \emptyset$.

(2) For each $i \in [n]$, since v_i and $u_{i,1}$ are adjacent in H, $|C(\Gamma) \cap \{v_i, u_{i,1}\}| \le 1$. If $C(\Gamma) \cap \{v_i, u_{i,1}\} = \emptyset$, then all vertices in $\{v_i\} \cup U_i \cup W_i$ are exposed. It follows that $\Gamma' = \Gamma \cup \{S(u_{i,1}; \{v_i\} \cup W_i)\}$ ia a t^+-star packing of H larger than Γ. Thus we conclude that $|C(\Gamma) \cap \{v_i, u_{i,1}\}| = 1$ for $i = 1, 2, \ldots, n$.

(3) Let e_j be any edge in $E(G)$ with end vertices v_i and v_s. If $e_j \notin V(\Gamma)$, then both $u_{i,1}$ and $u_{s,1}$ are in $C(\Gamma)$ by (2). Let $\Gamma' = (\Gamma \setminus \{S(u_{i,1}; \{v_i\} \cup W_i)\}) \cup \{S(v_i; \{e_j\} \cup U_i)\}$. It is clear that $|V(\Gamma)| = |V(\Gamma')|$ and $|V(\Gamma') \cap E(G)| > |V(\Gamma) \cap E(G)|$, contradicting the choice of Γ. Thus $E(G) \subseteq V(\Gamma)$.

□

Now let Γ be a maximum t^+-star packing of H such that $|V(\Gamma) \cap E(G)|$ is as large as possible. Let $K = C(\Gamma) \cap V(G)$. Since $E(G) \subseteq V(\Gamma)$ by Claim (3) and each e_j in $E(G)$ can only be covered by some star with its center in K, we conclude that K is a vertex cover of G. On the other hand, since

$$\begin{aligned} |V(\Gamma)| &= m + |K|t + (n - |K|)(t+1) \\ &= m + n(t+1) - |K| \\ &\ge p = m + n(t+1) - k\,, \end{aligned}$$

we have $|K| \le k$. Thus K is a vertex cover of G of cardinality at most k.

Now the NP-completeness of t^+-Star Packing in bipartite graphs follows from that of Vertex Cover. □

The minimum vertex cover problem in cubic graphs remains NP-complete [4]. Note that, in the proof of Theorem 3.1, if $t = 2$ and G is a cubic graph then the graph H constructed from G is a bipartite graph with maximum degree 4. Thus we immediately have the following corollary.

Corollary 3.2. *The problem 2^+-Star Packing is NP-complete in bipartite graphs with maximum degree* 4.

4. Approximation algorithms for $T \setminus t$-star packing and 2^+-star packing of graphs

Let T be a positive integer. A $\{K_{1,1}, K_{1,2}, \ldots, K_{1,T}\}$-packing a called a *$T$-star packing.* A $\{K_{1,T}, K_{1,T+1}, \ldots\}$-packing a called a *$T^+$-star packing.* Let T and t be two positive integers with $t < T$. A $\{K_{1,1}, K_{1,2}, \ldots, K_{1,T}\} \setminus \{K_{1,t}\}$-packing a called a *$T \setminus t$-star packing.*

In this section, we use different strategies to design approximation algorithms for maximum $T \setminus t$-star packing (with $t \geq 2$) and 2^+-star packing of general graphs. In Algorithm 3, we first find an optimal T-star packing of the input graph and then modify it into a $T \setminus t$-star packing. While in Algorithm 4, we first find a rooted spanning tree $T(r)$ of the input graph G and then find a 2^+-star packing of $T(r)$ and use it as an approximation solution.

Algorithm 3. Algorithm for $T \setminus t$-star packing.

```
Input: a graph $G$ and two positive integers $T$ and $t$ with $2 \leq t < T$
Output: a $T \setminus t$-star packing of $G$
1: find a maximum $T$-star packing $\Gamma$ of $G$
2: let $\Gamma = \{H_1, H_2, \ldots, H_k\}$
3: for $i := 1$ to $k$ do
4:     if $|V(H_i)| = t + 1$ then
5:         delete one leaf from $H_i$
6:     end if
7: end for
8: return $\{H_1, H_2, \ldots, H_k\}$
```

For a graph G, we use $\text{opt}(G, T)$ (resp. $\text{opt}(G, T \setminus t)$, $\text{opt}(G, 2^+)$) denote the number of vertices covered by a maximum T-star packing (resp. $T \setminus t$-star packing, 2^+-star packing) of G.

Theorem 4.1. *Let T and t be two positive integers with $2 \leq t < T$. Algorithm 3 finds a $T \setminus t$-star packing of the input graph G which covers at least $\frac{t}{t+1}\text{opt}(G, T \setminus t)$ vertices of G. The running time is $\mathcal{O}(m\sqrt{n})$, where m is the number of edges and n is the number of vertices of G.*

Proof. Algorithm 3 first finds a maximum T-star packing $\Gamma = \{H_1, H_2, \ldots, H_k\}$ of the input graph G. This can be done by applying an algorithm designed in [2] within $\mathcal{O}(m\sqrt{n})$ time steps. If some H_i is a t-star then the algorithm deletes one vertex from it and obtains a $(t-1)$-star of G. This clearly spends at most $\mathcal{O}(n)$ time steps. Thus the overall running time of the algorithm is $\mathcal{O}(m\sqrt{n})$. It is also obvious that the resulting $T \setminus t$-star packing of G covers at least $\frac{t}{t+1}\text{opt}(G, T) \geq \frac{t}{t+1}\text{opt}(G, T \setminus t)$ vertices of G. □

Let $T(r)$ be a tree rooted at r. Let v be any vertex of $T(r)$. Recall that, in Section 2, we define the notations $l(v)$, $T(v)$, $T^*(v)$, $F(v)$, $\text{GF}(v)$, $B[v]$ and $\text{CHD}(v)$.

Algorithm 4. Algorithm for 2^+-star packing.

```
Input: a connected graph $G$
Output: a $2^+$-star packing of $G$
1: find a rooted spanning tree $T(r)$ of $G$
2: set $H := T(r)$ and $\Gamma := \emptyset$
3: while $|V(H)| \geq 3$ do
4:     while there is a leaf $v$ of $H$ with highest level that has siblings do
5:         find a leaf $v$ of $H$ with highest level
6:         if $|V(H) \setminus T(F(v))| = 0$ or $|V(H) \setminus T(F(v))| \geq 3$ then
```

```
            set $\Gamma := \Gamma \cup \{S_H(F(v); B[v])\}$ and $H := H - T(F(v))$
        else
            set $\Gamma := \Gamma \cup \{S_H(F(v))\}$ and $H := H - T^*(F(v))$
        end if
    end while
    find a leaf $v$ of $H$ with highest level
    if $F(v)$ has siblings then
        if $|V(H) \setminus T(\mathrm{GF}(v))| = 0$ or $|V(H) \setminus T(\mathrm{GF}(v))| \geq 3$ then
            set $\Gamma := \Gamma \cup \{S_H(\mathrm{GF}(v); B[F(v)])\}$ and $H := H - T(\mathrm{GF}(v))$
        else
            set $\Gamma := \Gamma \cup \{S_H(\mathrm{GF}(v))\}$ and $H := H - T^*(\mathrm{GF}(v))$
        end if
    else
        set $\Gamma := \Gamma \cup \{S_H(F(v))\}$ and $H := H - \{v, F(v), \mathrm{GF}(v)\}$
    end if
end while
return $\Gamma$
```

Theorem 4.2. *Let G be a connected graph with at least three vertices. Algorithm 4 finds a 2^+-star packing of G that covers at least $\frac{1}{2}|V(G)|$ vertices. The running time is $\mathcal{O}(m)$, where m is the number of edges of G.*

Proof. Suppose H is the current tree dealt with by the algorithm at some stage of the execution. If $|V(H)| \geq 3$, then it is going to find a star, say S. Let H' be the tree obtained from H by deleting some vertices immediately after S is selected. Then it is not difficult to see that in any cases we have $|V(S)| \geq \frac{|V(H)|-|V(H')|}{2}$. If S is the last star the algorithm selects, then $|V(H')| \leq 2$ and it is easy to check that in any cases $|V(S)| \geq \frac{|V(H)|}{2}$. Thus in overall the algorithm outputs a 2^+-star packing of G that covers at least $\frac{1}{2}|V(G)|$ vertices.

It spends at most $\mathcal{O}(m)$ time steps to find a spanning tree of G. The process that the algorithm selects stars will be completed within time $\mathcal{O}(n)$. Thus the running time of the algorithm is $\mathcal{O}(m)$. □

It is worth to point out that the lower bound $\frac{1}{2}|V(G)|$ in this theorem is sharp. Let G be a tree obtained from a star by adding a new vertex for each leaf and an edge between them. It is not difficult to see that the maximum 2^+-star packing of G covers exactly $\frac{1}{2}(|V(G)|+1)$ vertices.

5. Conclusion & future work

Hell and Kirkpatrick [8] proved that the $\mathcal{H}$-factor problem is NP-complete unless $\mathcal{H}$ is a sequential star set and designed a polynomial-time algorithm for the $\{K_{1,1}, K_{1,2}, \ldots, K_{1,k}\}$-packing problem in general graphs. We investigate star family packing which is not a sequential star set. We design a linear-time algorithm to compute the maximum I-star packing of trees and prove that the t^+-star packing problem for $t \geq 2$ is NP-hard in bipartite graphs. We also design approximation algorithms for the $T \setminus t$-star packing problem and the 2^+-star packing problem.

As we can see, there is little research on star family packing of graphs in the literature. It will be meaningful to design algorithms for certain star family packing of some special classes of graphs. In particular, it is very important to investigate the cases when the star family consists of a single or two small stars.

Acknowledgements. We would like to give our thanks to anonymous reviewers for careful reading of this paper and many valuable suggestions. Supported by NSFC 11771080.

References

[1] R. Anstee, Personal Communication to Pavol Hell (1996).

[2] M. Bahenko and A. Gusakov, New exact and approximation algorithms for the star packing problem in undirected graphs. In: 28th International Symposium on Theoretical Aspects of Computer Science, STACS 2011. March 10–12, 2011, Dortmund, Germany (2011) 519–530.

[3] R.C. Brewster, P. Hell and R. Rizzi, Oriented star packings. *J. Combin. Theory, Ser. B* **98** (2008) 558–576.

[4] M. Chlebík and J. Chlebíková, Complexity of approximating bounded variants of optimization problems. *Theoret. Comput. Sci.* **354** (2006) 320–338.

[5] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, San Francisco (1979).

[6] D. Hartvigsen, P. Hell and J. Szabó, The k-piece packing problem. *J. Graph Theory* **52** (2006) 267–293.

[7] P. Hell and D.G. Kirkpatrick, Packing by cliques and by finite families of graphs. *Discrete Math.* **49** (1984) 45–59.

[8] P. Hell and D.G. Kirkpatrick, Packing by complete bipartite graphs. *SIAM J. Alg. Disc. Meth.* **7** (1986) 199–209.

[9] P. Hell, D.G. Kirkpatrick and J. Kratochvíl and I. Kříž, On restricted two-factors. *SIAM J. Discrete Math.* **1** (1988) 472–484.

[10] A. Kelmans, Optimal packing of induced stars in a graph. *Discrete Math.* **173** (1997) 97–127.

[11] M. Loebl and S. Poljak, Efficient subgraph packing. *J. Combin. Theory Ser. B* **59** (1993) 106–121.

[12] Q. Ning, On the star packing problem. In: Vol. 576 of *Proc. 1st China-USA International Graph Theory Conference* (1989) 411–416.