# The $P_k$ partition problem and related problems in bipartite graphs

*Jérôme Monnot* [*] and *Sophie Toulouse* [†]

**Abstract**

In this paper, we continue the investigation proposed in [15] about the approximability of $\mathbf{P}_k$ partition problems, but focusing here on their complexity. More precisely, we prove that the problem consisting of deciding if a graph of $nk$ vertices has $n$ vertex disjoint simple paths $\{P_1, \cdots, P_n\}$ such that each path $P_i$ has $k$ vertices is **NP**-complete, even in bipartite graphs of maximum degree 3. Note that this result also holds when each path $P_i$ is chordless in $G[V(P_i)]$. Then, we prove that MAX$\mathbf{P}_3$PACKING and MAXINDUCED$\mathbf{P}_3$PACKING in bipartite graphs of maximum degree 3 are not in **PTAS**. Finally, we propose a 3/2-approximation for MIN3-PATHPARTITION in general graphs within $O(nm + n^2 \log n)$ time and a 1/3 (resp., 1/2)-approximation for MAXW$\mathbf{P}_3$PACKING in general (resp., bipartite) graphs of maximum degree 3 within $O(\alpha(n, 3n/2)n)$ (resp., $O(n^2 \log n)$) time, where $\alpha$ is the inverse Ackerman's function and $n = |V|$, $m = |E|$.

**Keywords:** $P_k$-partition; Induced $P_k$-partition; maximum (weighted) $P_k$-packing; maximum (weighted) induced $P_k$-packing; minimum $k$-path partition; bipartite graphs; **NP**-complete; **APX**-hard; approximation algorithms.

## 1 Introduction

The $\mathbf{P}_k$ partitioning problem ($\mathbf{P}_k$PARTITION in short) consists, given a simple graph $G = (V, E)$ on $k \times n$ vertices, of deciding if there exists a partition of $V$ into $(V_1, \cdots, V_n)$ such that for $1 \leq i \leq n$, $|V_i| = k$ and the subgraph $G[V_i]$ induced by $V_i$ contains a Hamiltonian path. In other words, we want to know if there exists $n$ vertex disjoint simple paths of length $k$ in $G$. The analogous problem where the subgraph $G[V_i]$ induced by $V_i$ is isomorphic to $\mathbf{P}_k$ (the chordless path on $k$ vertices) will be denoted by INDUCED $\mathbf{P}_k$PARTITION. These two problems are **NP**-complete for any $k \geq 3$, and polynomial otherwise, [8, 13]. In fact, they both are a particular case of a more general problem called *partition into isomorphic subgraphs*, [8]. In [13], Kirkpatrick and Hell give a necessary and sufficient condition for the **NP**-completeness of the partition into isomorphic subgraphs problem in general graphs.

$\mathbf{P}_k$PARTITION has been widely studied in the literature, mainly because its **NP**-completeness also implies the **NP**-hardness of two famous optimization problems, namely: the minimum $k$-path partition problem (denoted by MIN$k$-PATHPARTITION) and the maximum $\mathbf{P}_k$ packing problem (MAX$\mathbf{P}_k$PACKING in short). MIN$k$-PATHPARTITION consists of partitioning

[*]CNRS LAMSADE - UMR 7024, Université Paris-Dauphine, Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16, France. `monnot@lamsade.dauphine.fr`

[†]LIPN - UMR CNRS 7030, Institut Galilée, Université Paris 13, 99 av. Jean-Baptiste Clément, 93430 Villetaneuse, France. `sophie.toulouse@lipn.univ-paris13.fr`

the vertex set of a graph $G = (V, E)$ into the smallest number of paths such that each path has *at most* $k$ vertices (for instance, MIN2-PATHPARTITION is equivalent to the edge cover problem); the optimal value is usually denoted by $\rho_{k-1}(G)$, and by $\rho(G)$ when no constraint occurs on the length of the paths (in particular, we have $\rho(G) = 1$ *iff* $G$ has an Hamiltonian path). MIN$k$-PATHPARTITION has been extensively studied in the literature, [18, 17, 21], and has applications in broadcasting problems, see for example [21]. MAX$\mathbf{P}_k$PACKING (resp., MAXINDUCED$\mathbf{P}_k$PACKING), consists, given a simple graph $G = (V, E)$, of finding a maximum number of vertex-disjoint (resp., induced) $\mathbf{P}_k$. In their weighted versions (denoted MAXW$\mathbf{P}_k$PACKING and MAXWINDUCED$\mathbf{P}_k$PACKING, respectively), the input graph $G = (V, E)$ is given together with a weight function $w : E \rightarrow \mathbb{N}$ on its edges; the goal is to find a collection $\mathcal{P} = \{P_1, \ldots, P_q\}$ of vertex-disjoint (resp., induced $\mathbf{P}_k$) maximizing $w(\mathcal{P}) = \sum_{i=1}^{q} \sum_{e \in P_i} w(e)$. Some approximation results for MAXW$\mathbf{P}_k$PACKING when the graph is complete on $k \times n$ vertices are given in [9, 10, 15]. In this case, each solution contains exactly $n$ vertex disjoints paths of length $k - 1$ (note that, in this particular case, the minimization version may also be considered). This problem is related to the vehicle routing problem, [21, 3].

Here, we study the complexity of $\mathbf{P}_k$PARTITION and INDUCED $\mathbf{P}_k$PARTITION in the case of bipartite graphs. We first show that $\mathbf{P}_k$PARTITION and INDUCED $\mathbf{P}_k$PARTITION are **NP**-complete for any $k \geq 3$ in bipartite graphs of maximum degree 3. Moreover, for $k = 3$, this remains true even if the graph is planar. On the opposite, $\mathbf{P}_k$PARTITION, INDUCED $\mathbf{P}_k$PARTITION, MIN$k$-PATHPARTITION and MAXW$\mathbf{P}_k$PACKING trivially become polynomial-time computable in graphs of maximum degree 2 and in forests. Then, we prove that, in bipartite graphs of maximum degree 3, MAX$\mathbf{P}_k$PACKING and MAXINDUCED$\mathbf{P}_k$PACKING are not in **PTAS**. More precisely, we prove that there is a constant $\varepsilon_k > 0$ such that it is **NP**-hard to decide whether a maximum (induced) $\mathbf{P}_k$-packing of a bipartite graph of maximum degree 3 on $kn$ vertices is of size $n$ or of size upper bounded by $(1 - \varepsilon_k)n$. Finally, we propose a 3/2-approximation for MIN3-PATHPARTITION in general graphs and a 1/3 (resp., 1/2)-approximation for MAXW$\mathbf{P}_3$PACKING in general (resp., bipartite) graphs of maximum degree 3.

This paper is organized as follows: in the next section, we briefly present previous related works about the hardness of solving bounded-size-path packing problems. Then, the third part is dedicated to complexity results concerning $\mathbf{P}_k$PARTITION, INDUCED $\mathbf{P}_k$PARTITION, MAXINDUCED$\mathbf{P}_k$PACKING and MAX$\mathbf{P}_k$PACKING in bipartite graphs. Finally, some approximation results concerning MAXW$\mathbf{P}_3$PACKING and MIN3-PATHPARTITION are proposed in a fourth section.

The notations are the usual ones according to graph theory. Moreover, we exclusively work in undirected simple graphs. In this paper, we often identify a path $P$ of length $k - 1$ with $\mathbf{P}_k$, even if $P$ contains a chord. However, when we deal with INDUCED $\mathbf{P}_k$PARTITION, the paths considered will be chordless. We denote by $opt(I)$ and $apx(I)$ the value of an optimal and of an approximate solution, respectively. We say that an algorithm $\mathcal{A}$ is an $\varepsilon$-approximation with $\varepsilon \geq 1$ for a minimization problem (resp., with $\varepsilon \leq 1$ for a maximization problem) if $apx(I) \leq \varepsilon \times opt(I)$ (resp., $apx(I) \geq \varepsilon \times opt(I)$) for any instance $I$ (for more details, see for instance [2]).

## 2 Previous related work

The minimum $k$-path partition problem is obviously **NP**-complete in general graphs [8], and remains intractable in comparability graphs, [18], in cographs, [17], and in bipartite chordal graphs, [18] (when $k$ is part of the input). Note that most of the proofs of **NP**-completeness actually establish the **NP**-completeness of $\mathbf{P}_k$Partition. Nevertheless, the problem turns out to be polynomial-time solvable in trees, [21], in cographs when $k$ is fixed, [17] or in bipartite permutation graphs, [18]. Note that one can also find in the literature several results about partitioning the graph into disjoints paths of length at least 2, [19, 11].

Concerning the approximability of related problems, Hassin and Rubinstein, [9] proposed a generic algorithm to approximate MaxW$\mathbf{P}_4$Packing in complete graphs on $4n$ vertices that guarantees an approximation ratio of 3/4 for general distance function. More recently in [15], it has been proven that this algorithm is also a 9/10-approximation for the $1,2$-instances. For the minimization version, it provides respectively a 3/2- and a 7/6-approximation for the metric and the $1,2$- instances in complete graphs on $4n$ vertices (in this case, we seek a maximal $\mathbf{P}_4$-Packing of minimum weight). In [10], the authors proposed a $(35/67-\varepsilon)$-approximation for Max$\mathbf{P}_3$Partition in complete graphs on $3n$ vertices using a randomized algorithm. To our knowledge, there is no specific approximation results for MaxW$\mathbf{P}_3$Packing in general graphs. However, using approximation results for the maximum weighted 3-packing problem (mainly based on local search techniques), [1], we can obtain a $(\frac{1}{2}-\varepsilon)$-approximation for MaxW$\mathbf{P}_3$Packing. Finally, there is, to our knowledge, no approximation result for Min$k$-PathPartition. Nevertheless, when the problem consists of maximizing the number of edges used by the paths, then we can find some approximation results, in [20] for the general case, in [5] for dense graphs.

## 3 Complexity results

**Theorem 3.1** $\mathbf{P}_k$Partition *and* induced $\mathbf{P}_k$Partition *are* ***NP****-complete in bipartite graphs of maximum degree 3, for any* $k \geq 3$. *As a consequence,* Max$\mathbf{P}_k$Packing *and* Min$k$-PathPartition *are* ***NP****-hard in bipartite graphs with maximum degree 3, for any* $k \geq 3$.

**Proof:** The paths of length $k-1$ used in the reduction are chordless; thus, the result holds for both problems. The proof is based on a reduction from the $k$-dimensional matching problem, denoted by $k$DM, which is known to be **NP**-complete, [8]. An instance of $k$DM consists of a subset $\mathcal{C} = \{c_1, \ldots, c_m\} \subseteq X_1 \times \ldots \times X_k$ of $k$-tuples, where $X_1, \ldots, X_k$ are $k$ pairwise disjoint sets of size $n$. A matching is a subset $M \subseteq \mathcal{C}$ such that no element in $M$ agrees in any coordinate, and the purpose of $k$DM is to answer the question: does there exist a perfect matching $M$ on $\mathcal{C}$, that is, a matching of size $n$?

Given an instance $I = (\mathcal{C}, X_1 \times \ldots \times X_k)$ of $k$DM, we build an instance $G = (V, E)$ of $\mathbf{P}_k$Partition depending on the parity of $k$, where $G$ is a bipartite graph of maximum degree 3, as follows:

**case 1**: $k$ is odd.

• To each $k$-tuple $c_i \in \mathcal{C}$, we associate a gadget $H(c_i)$ that consists of a collection $\left\{P^{i,1}, \ldots, P^{i,k}\right\}$ of $k$ vertex-disjoint $\mathbf{P}_k$ with $P^{i,q} = \left\{a_1^{i,q}, \ldots, a_k^{i,q}\right\}$ for $q = 1, \ldots, k$. We add to $H(c_i)$ the edges $[a_1^{i,q}, a_1^{i,q+1}]$ for $q = 1$ to $k-1$, in order to form a $(k+1)$-th $\mathbf{P}_k$ $\left\{a_1^{i,1}, \ldots, a_1^{i,k}\right\}$ (see Figure 1 for an illustration when $k = 3$).

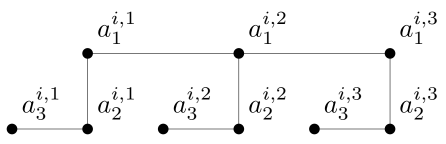


Figure 1: The gadget $H(c_i)$ when $c_i$ is a 3-uplet.

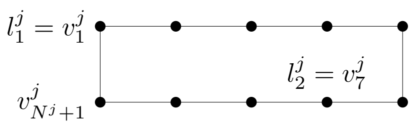


Figure 2: The gadget $H(e_j)$ for $k = 3$ and $d^j = 2$.

• For each element $e_j \in X_1 \cup \ldots \cup X_k$, let $d^j$ denote the number of $k$-tuples $c_i \in \mathcal{C}$ that contain $e_j$; the gadget $H(e_j)$ is defined as a cycle $\left\{v_1^j, \ldots, v_{N^j+1}^j, v_1^j\right\}$ on $N^j + 1$ vertices, where $N^j = k(2d^j - 1)$. Moreover, for $p = 1$ to $d^j$, we denote by $l_p^j$ the vertex of index $2k(p-1)+1$ (see Figure 2 for an illustration of $H(e_j)$ when $k = 3$ and $d^j = 2$).

• Finally, for any couple $(e_j, c_i)$ such that $e_j$ is the value of $c_i$ on the $q$-th coordinate, the two gadgets $H(c_i)$ and $H(e_j)$ are connected using an edge $[a_2^{i,q}, l_{p_i}^j]$. The vertices $l_{p_i}^j$ that will be linked to a given gadget $H(c_i)$ must be chosen in such a way that each vertex $l_p^j$ from any gadget $H(e_j)$ will be connected to exactly one gadget $H(c_i)$ (this is possible since each $H(e_j)$ contains exactly $d^j$ vertices $l_p^j$).

This construction obviously leads to a graph $G$ of maximum degree 3, on $3k^2m + (1-k)kn$ vertices: consider, on the one hand, that each gadget $H(c_i)$ is a graph on $k^2$ vertices and, on the other hand, that $\sum_{j=1}^{kn} d^j = km$ (wlog., we assume that each element $e_j$ appears at least once in $\mathcal{C}$). Finally, one can see that $G$ is bipartite.

We claim that there exists a perfect matching $M \subseteq \mathcal{C}$ *iff* there exists a partition $\mathcal{P}^*$ of $G$ into $\mathcal{P}_k$. First, the following property can be easily proved:

**Property 3.2** *In any partition of $G$ into $\mathcal{P}_k$, and for any $i = 1, \ldots, m$, one uses either $\mathcal{P}^i$ or $\mathcal{Q}^i$, where $\mathcal{P}^i$ and $\mathcal{Q}^i$ are the collection of paths defined as:*

$$\forall i = 1, \ldots, m, \quad \forall q = 1, \ldots, k, \quad \left\{ \begin{array}{lcl} P^{i,q} & = & \left\{a_k^{i,q}, \ldots, a_2^{i,q}, l_{i,q}\right\} \\ Q^{i,q} & = & \left\{a_k^{i,q}, \ldots, a_2^{i,q}, a_1^{i,q}\right\} \end{array} \right.$$

*(where $l_{i,q}$ denotes the vertex from some $H(e_j)$ linked to $a_2^{i,q}$.*

$$\forall i = 1, \ldots, m, \qquad \left\{ \begin{array}{l} \mathcal{P}^i = \cup_{q=1}^k P^{i,q} \cup \left\{a_1^{i,1}, a_1^{i,2}, \ldots, a_1^{i,k}\right\} \\ \mathcal{Q}^i = \cup_{q=1}^k Q^{i,q} \end{array} \right.$$

Let $M$ be a perfect matching on $\mathcal{C}$; we build a packing $\mathcal{P}$ applying the following rule: if a given element $c_i$ belongs to $M$, then we use $\mathcal{P}^i$ to cover $H(c_i)$; we use $\mathcal{Q}^i$ otherwise. Figure 3 illustrates this construction for 3DM. Since $M$ is a perfect matching, exactly one vertex $l_p$ per gadget $H(e_j)$ is already covered by some $P^{i,q}$. Thus, on a given cycle $H(e_j)$, the $N^j = k(2d^j - 1)$ vertices that remain uncovered can easily be covered using a sequence of $(2d^j - 1)$ vertex disjoints $\mathcal{P}_k$.

Conversely, let $\mathcal{P}^* = \{P_1, \ldots, P_r\}$ be a partition of $G$ into $\mathbf{P}_k$; since each gadget $H(e_j)$ has $N^j = k(2d^j - 1) + 1$ vertices, at least one edge $e$ of some $P_\ell$ in $\mathcal{P}^*$ links $H(e_j)$ to

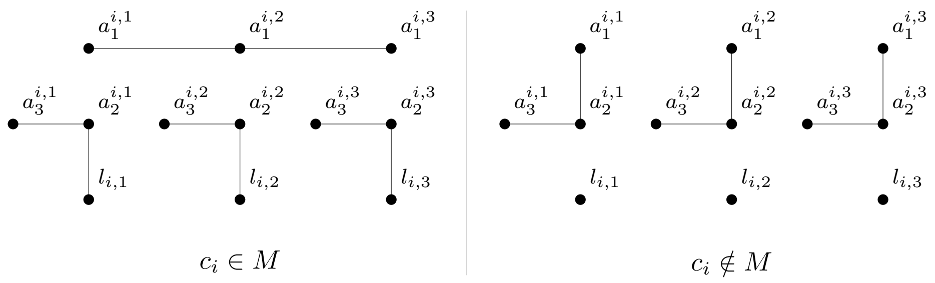


Figure 3: A vertex partition of a $H(c_i)$ gadget into 2-length paths.

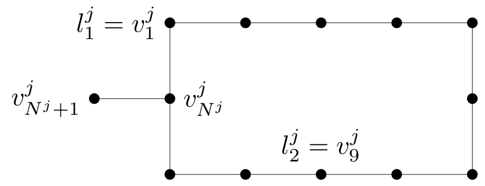


Figure 4: The gadget $H(e_j)$ for $k = 4$ and $d^j = 2$.

a given $H(c_i)$, using a $l_p$ vertex; we deduce from Property 3.2 that $P_\ell$ is some $P^{i,q}$ path and thus, that $l_p$ is the only vertex of $P_\ell$ that intersects $H(e_j)$. Consider now any two vertices $l_p$ and $l_{p'}$, $p < p'$, from $H(e_j)$; since $l_p = v_{2k(p-1)+1}$ and $l_{p'} = v_{2k(p'-1)+1}$, there are $2k(p' - p) - 1$ vertices between $l_p$ and $l_{p'}$, which might not be covered by any collection of $\mathcal{P}_k$. Hence, exactly one vertex from each $H(e_j)$ is covered by some $P^{i,q}$. Concerning $H(c_i)$, we already know that its vertices may be covered by either $\mathcal{P}^i$, or $\mathcal{Q}^i$. Hence, by setting $M = \{c_i \mid \mathcal{P}^i \subseteq \mathcal{P}^*\}$, we define a perfect matching, and the proof is complete.

**case 2**: $k$ is even.

The proof is quite identical, except the construction of the $H(e_j)$ gadgets: $H(e_j)$ is a cycle $\left\{v_1^j, \ldots, v_{N^j}^j, v_1^j\right\}$ on $N^j$ vertices, plus an additional edge $[v_{N^j}^j, v_{N^j+1}^j]$ (see Figure 3 for an illustration when $k = 4$ and $d^j = 2$). The special vertices $l_p^j$ are defined by $l_p^j = v_{2k(p-1)+1}^j$ for $p = 1, \cdots, d^j$ (note that $l_{d^j}^j$ never matches $v_{N^j}^j$). We can easily see that $H(e_j)$ is bipartite ($N^j$ is even). Moreover, if one vertex $l_p^j$ is missing (that is, covered by a path having its vertices but $l_p^j$ in some $H(c_i)$ gadget), then $V(H(e_j))\backslash \left\{l_p^j\right\}$ can be covered by a collection of vertex disjoints $\mathbf{P}_k$. Finally, if two vertices $l_p^j$ and $l_{p'}^j$ are already covered, then $V(H(e_j))\backslash \left\{l_p^j, l_{p'}^j\right\}$ may not be covered by a collection of vertex disjoints $\mathbf{P}_k$.

■

If we decrease the maximum degree of the graph down to 2, we can easily prove that $\mathbf{P}_k$PARTITION, INDUCED $\mathbf{P}_k$PARTITION, MAX$\mathbf{P}_k$PACKING and MIN$k$-PATHPARTITION are polynomial-time computable. The same fact holds for MAXW$\mathbf{P}_k$PACKING, although it is a little bit complicated. Moreover, this result holds in forests.

**Proposition 3.3** MAXW$\mathbf{P}_k$PACKING *is polynomial in graphs with maximum degree 2 and in forests, for any* $k \geq 3$.

**Proof:** See appendix. ■

**Theorem 3.4** $\mathbf{P}_3$PARTITION *and* INDUCED $\mathbf{P}_3$PARTITION *are* ***NP****-complete in planar bipartite graphs with maximum degree 3. As a consequence,* MAX$\mathbf{P}_3$PACKING *and* MIN3-PATHPARTITION *are* ***NP****-hard in planar bipartite graphs with maximum degree 3.*

**Proof:** We apply the proof of the previous theorem, except that we start from a restriction of the 3-dimensional matching problem, which is denoted by PLANAR 3DM-3. With respect to this restriction, on the one hand, each element $e_j \in X_1 \cup X_2 \cup X_3$ appears in at most three distinct 3-tuples $c_i \in \mathcal{C}$ and, on the other hand, the characterization bipartite graph $G(\mathcal{C})$ of the instance is planar. The left-hand-side and the right-hand-side vertex sets of $G(\mathcal{C})$ respectively represent the 3-tuples from $\mathcal{C}$ and the elements from $X_1 \cup X_2 \cup X_3$; thus, a left vertex $l_i$ will be linked to a right one $r_j$ *iff* the corresponding 3-tuple $c_i$ contains the corresponding element $e_j$. It is well known that this restriction of 3DM is still **NP**-complete, [6]. In order to apply the previous construction properly, we have to link the $H(c_i)$ gadgets to the $H(e_j)$ gadgets in such a way that the final graph $G$ is planar. As a consequence, for any couple $(H(c_i), H(e_j))$ such that $e_j \in c_i$, the choice of the vertex $l_p^j$ from $H(e_j)$ that will be linked to $H(c_i)$ is no longer free, but depends on the characteristic graph $G(\mathcal{C})$ of the input instance. ■

Using an **APX**-hardness result for the optimization version of $k$DM (denoted MAX$k$DM) and the reduction of Theorem 3.1, we are able to obtain an **APX**-hardness result for MAX$\mathbf{P}_k$PACKING in bipartite graphs of maximum degree 3. The result used is the following: For any $k \geq 3$, there is a constant $\varepsilon'_k > 0$, such that $\forall I = (\mathcal{C}, X_1 \times \ldots \times X_k)$ instance of MAX$k$DM with $n = |X_1| = \cdots = |X_k|$, it is **NP**-hard to decide between $opt(I) = n$ and $opt(I) \leq (1 - \varepsilon'_k)n$, where $opt(I)$ is the value of a maximum matching on $\mathcal{C}$. This result also holds if we restrict us to instances $I = (\mathcal{C}, X_1 \times \ldots \times X_k)$ such that for each element $e_j \in X_1 \cup \ldots \cup X_k$, $d^j \leq f(k)$, where $f(k)$ is a constant (we recall that $d^j$ is the number of $k$-tuples $c_i \in \mathcal{C}$ containing $e_j$). For $k = 3$, the result is proved in [16] with $f(3) = 3$, and for the other values of $k$, [12].

**Theorem 3.5** *For any $k \geq 3$, there is a constant $\varepsilon_k > 0$, such that $\forall G = (V, E)$ instance of* MAX$\mathbf{P}_k$PACKING *(resp.,* MAXINDUCED$\mathbf{P}_k$PACKING*) where $G$ is a bipartite graph of maximum degree 3, it is* ***NP****-hard to decide between $opt(G) = \frac{|V|}{k}$ and $opt(G) \leq (1 - \varepsilon_k)\frac{|V|}{k}$ where $opt(G)$ is the value of a maximum (resp., maximum induced)* $\mathbf{P}_k$*-Packing on $G$.*

**Proof:** Let $I = (\mathcal{C}, X_1 \times \ldots \times X_k)$ be an instance of MAX$k$DM with $n = |X_1| = \cdots = |X_k|$ and $m = |\mathcal{C}|$ and such that $\forall e_j \in X_1 \cup \ldots \cup X_k$, $d^j \leq f(k)$. Consider the graph $G = (V, E)$ produced in Theorem 3.1. We recall that $G$ is bipartite of maximum degree 3, $|V| = 3k^2 m + (1 - k)n$, and all paths of length $k - 1$ are chordless. Let $\mathcal{P}^*$ be an optimal solution of MAX$\mathbf{P}_k$PACKING with value $opt(G)$. Let us prove that we can assume that:

$(i)$ In any gadget $H(c_i)$, $\mathcal{P}^*$ contains either the packing $\mathcal{P}^i$, or the packing $\mathcal{Q}^i$.

$(ii)$ In any gadget $H(e_j)$, $\mathcal{P}^*$ contains exactly $2d^j - 1$ paths.

For $(i)$, any optimal solution must use (at least) one of the two vertices $a_1^{i,q}$ and $l_{i,q}$, for any couple $(i, q)$ (where we recall that $l_{i,q} = l_{p_i}^j$ is a vertex from some $H(e_j)$ linked to $a_2^{i,q}$). Suppose the reverse for some $(i, q)$: then, none of the vertices $l_{i,q}, a_1^{i,q}, a_2^{i,q}, \ldots, a_k^{i,q}$ may be part of a path from $\mathcal{P}^*$ and thus, $P^{i,q}$ or $Q^{i,q}$ could be added to $\mathcal{P}^*$. Hence, if the edge $[a_1^{i,q}, a_2^{i,q}]$ (resp., $[a_2^{i,q}, l_{i,q}]$ and not $[a_1^{i,q}, a_2^{i,q}]$) is used by some path $P$ of $\mathcal{P}^*$, $P$ can be replaced in $\mathcal{P}^*$ by the path $Q^{i,q}$ (resp., $P^{i,q}$). If none of the edges $[a_1^{i,q}, a_2^{i,q}]$ and $[a_2^{i,q}, l_{i,q}]$ are used by $\mathcal{P}^*$, replace by $P^{i,q}$ the path from $\mathcal{P}^*$ that uses $l_{i,q}$ if such a path exists, replace by $Q^{i,q}$ the path from $\mathcal{P}^*$ that uses $a_1^{i,q}$ otherwise. Moreover, if $\left\{a_1^{i,1}, a_1^{i,2}, \ldots, a_1^{i,k}\right\} \notin \mathcal{P}^*$, then $k$ paths of $\mathcal{P}^*$ are in $H(c_i)$ ($\mathcal{P}^*$ is an optimal solution). Hence, we can replace each path $P^{i,q}$ by $Q^{i,q}$ and in this case, $\mathcal{P}^*$ contains the packing $\mathcal{Q}^i$. Now, assume that $\left\{a_1^{i,1}, a_1^{i,2}, \ldots, a_1^{i,k}\right\} \in \mathcal{P}^*$. If $\mathcal{P}^*$ does not use the $k$ paths $P^{i,q}$ (ie., the packing $\mathcal{P}^i$), then by deleting the paths $\left\{a_1^{i,1}, a_1^{i,2}, \ldots, a_1^{i,k}\right\}$, $P^{i,q}$ and by adding the $k$ paths $Q^{i,q}$, we obtain another optimal solution.

For $(ii)$, assume the reverse. Since $\mathcal{P}^*$ is an optimal solution, at least 2 vertices $l^j_{p_i}$ and $l^j_{p_{i'}}$ of $H(e_j)$ are used in $\mathcal{P}^*$ by paths $P^{i,q}$ and $P^{i',q'}$ with $p_i < p_{i'}$. Choose two consecutive such vertices, in the sense that $\mathcal{P}^*$ does not use any of the paths $P^{i'',q''}$ for $l^j_{p_{i''}}$ such that $p_i < p_{i''} < p_{i'}$. Now, since there are $2k(p_{i'} - p_i) - 1$ vertices of $H(e_j)$ between $l^j_{p_i}$ and $l^j_{p_{i'}}$, we can replaced $P^{i,q}$, $P^{i',q'}$ and the paths of $\mathcal{P}^*$ between vertices $l^j_{p_i}$ and $l^j_{p_{i'}}$ by $P^{i,q}$ and $2(p_{i'} - p_i)$ paths using vertices between $l^j_{p_i}$ and $l^j_{p_{i'}}$, plus $l^j_{p_{i'}}$ (in this case, observe that the packing $\mathcal{P}^{i'}$ will be replaced by the packing $\mathcal{Q}^{i'}$ according to property $(i)$). Thus, by repeating this procedure, the result follows.

We know that $I$ has a perfect matching *iff* $opt(G) = 3km + (1-k)n = \frac{|V|}{k}$. Now, let $M_0 = \{c_i \in \mathcal{C} : \ \mathcal{P}^* \text{ contains the packing } \mathcal{P}^i \text{ on } V(H(c_i))\}$ and $m_0 = |M_0|$. Since $\sum_{j=1}^k nd^j = km$ (see Theorem 3.1), and using properties $(i)$ and $(ii)$, we deduce $opt(I) = 2km - kn + km + m_0$. Thus, if a maximum matching on $I$ for MAX$k$DM with value $opt(I)$ verifies $opt(I) \leq (1 - \varepsilon'_k)n$, we deduce $m_0 \leq (1-\varepsilon'_k)n$. Hence, by setting $\varepsilon_k = \frac{n}{3km - kn + n}\varepsilon'_k$, we obtain $opt(G) \leq (1-\varepsilon_k)(3km - kn + n) = (1-\varepsilon_k)\frac{|V|}{k}$. Finally, since $d^j \leq f(k)$ where $f(k)$ is a constant, we deduce that $km \leq 3f(k)n$ and then, $\varepsilon_k \geq \frac{1}{9f(k)+1-k}\varepsilon'_k$. The proof is now complete.

■

Some interesting questions concern the complexity of $\mathbf{P}_k$PARTITION (or INDUCED $\mathbf{P}_k$PARTITION) for $k \geq 4$ and the **APX**-hardness of MAX$\mathbf{P}_k$PACKING and MAXINDUCED$\mathbf{P}_k$PACKING (or MAXINDUCED$\mathbf{P}_k$PACKING) for $k \geq 3$ in planar bipartite graphs with maximum degree 3.

## 4 Approximation results

We present some approximation results for MAXW$\mathbf{P}_3$PACKING and MIN3-PATHPARTITION, that are mainly based on matching and spanning tree heuristics.

### 4.1 MaxWP$_3$Packing in graphs of maximum degree 3

For this problem, the best approximate algorithm known so far provides a ratio of $(\frac{1}{2} - \varepsilon)$, within high (but polynomial) time complexity. This algorithm is deduced from the one proposed in [1] to approximate the weighted $k$-set packing problem for sets of size 3. Furthermore, a simple greedy $1/k$-approximation of MAXW$\mathbf{P}_k$PACKING consists of iteratively picking a path of length $k-1$ that is of maximum weight. For $k = 3$ and in graphs of maximum degree 3, the time complexity of this algorithm is between $O(n \log n)$ and $O(n^2)$ (depending on the encoding structure). Actually, in such graphs, one may reach a 1/3-approximate solution, even in time $O(\alpha(n,m)n)$, where $\alpha$ is the inverse Ackerman's function and $m \leq 3n/2$.

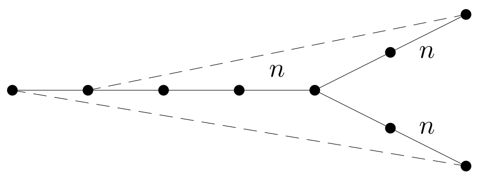


Figure 5: The tightness.

**Theorem 4.1** MaxW$\mathbf{P}_3$Packing *is 1/3 approximable within $O(\alpha(n, 3n/2)n)$ time complexity in graphs of maximum degree 3; this ratio is tight for the algorithm we analyse.*

**Proof:** We assume that the graph is connected (otherwise, we apply the same proof on each connected component containing at least 3 vertices). The argument lies on the following observation: for any spanning tree of maximum degree 3 containing at least 3 vertices, one can build a cover of its edge set into 3 packings of $\mathbf{P}_3$ within linear time (a formal proof is given in appendix). Hence, given a weighted connected graph $G = (V, E)$ of maximum degree 3, we compute a maximum-weight spanning tree $T = (V, E_T)$ on $G$. Because $G$ is of maximum degree 3, this can be done in $O(\alpha(n, 3n/2)n)$ time, [4]. We then compute $(\mathcal{P}^1, \mathcal{P}^2, \mathcal{P}^3)$ a $\mathbf{P}_3$-packing cover of $T$ and finally, pick the best $\mathbf{P}_3$-packing among $\mathcal{P}^1$, $\mathcal{P}^2$ and $\mathcal{P}^3$. The value of this packing is at least 1/3 times the weight of $T$, which is at least the weight of an optimal $P_3$-Packing on $G$, as any $P_3$-Packing can be extended into a spanning tree. The tightness of this algorithm is illustrated in Figure 5: the edges of $E_T$ are drawn in rigid lines, whereas the edges of $E \backslash E_T$ are drawn in dotted lines; finally, all the edges with no mention of their weight are of weight 1. Observe that an optimal $\mathbf{P}_3$-packing on $T$ is of weight $n + 3$, whereas $opt(I) = 3n + 3$. ■

## 4.2 MaxWP$_3$Packing in bipartite graphs of maximum degree 3

If we restrict us to bipartite graphs, we slightly improve the ratio of $\frac{1}{2} - \varepsilon$ ([1]) up to $\frac{1}{2}$. We then show that, in the unweighted case, this result holds without any constraint on the graph maximum degree.

From $I = (G, w)$ where $G$ is a bipartite graph $G = (L \cup R, E)$ of maximum degree 3, we build two weighted graphs $(G_L, d_L)$ and $(G_R, d_R)$, where $G_L = (L, E_L)$ and $G_R = (R, E_R)$. Two vertices $x \neq y$ from $L$ are linked in $G_L$ *iff* there exists in $G$ a path of length 2 $P_{x,y}$ from $x$ to $y$, rigorously: $[x, y] \in E_L$ *iff* $\exists z \in R$ s.t. $[x, z], [z, y] \in E$. The distance $d_L(x, y)$ is defined as $d_L(x, y) = \max\{w(x, z) + w(z, y) | [x, z], [z, y] \in E\}$. $(G_R, d_R)$ is defined by considering $R$ instead of $L$. If $G$ is of maximum degree 3, then the following fact holds:

**Lemma 4.2** *From any matching $M$ on $G_L$ (resp., on $G_R$), one can deduce a $\mathbf{P}_3$ packing $\mathcal{P}_M$ of weight $w(\mathcal{P}_M) = d_L(M)$ (resp., $w(\mathcal{P}_M) = d_R(M)$), when $G$ is of degree at most 3.)*

**Proof:** We only prove the result for $G_L$. Let $M$ be a matching on $G_L$. For any edge $e = [x, y] \in M$, there exists in $G$ a chain $P_e = \{x, z_e, y\}$ with $w(P_e) = d_L(e)$. Let us show that $\mathcal{P}_M = \{P_e | e \in M\}$ is a packing. Assume the reverse: then, there exists two edges $e_1 = [x_1, y_1]$ and $e_2 = [x_2, y_2]$ in $M$ such that $P_{e_1} \cap P_{e_2} \neq \emptyset$. Since $\{e_1, e_2\}$ is a matching, the four vertices $x_1$, $x_2$, $y_1$ and $y_2$ are pairwise distinct and then necessarily $z_{e_1} = z_{e_2}$. Hence, $z_{e_1}$ is linked to 4 vertices in $G$, which contradicts the fact that the maximum degree in $G$ does not exceed 3. ■

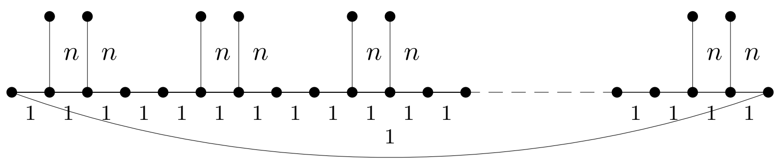


Figure 6: The tightness.

```
Weighted $P_3$-Packing
1 Build the weighted graphs $(G_L, d_L)$ and $(G_R, d_R)$;
2 Compute a maximum weight matching $M_L^*$ (resp., $M_R^*$) on $(G_L, d_L)$ (resp., on $(G_R, d_R)$);
3 Deduce from $M_L^*$ (resp., $M_R^*$) a $\mathbf{P}_3$ packing $\mathcal{P}_L$ (resp., $\mathcal{P}_R$) according to Lemma 4.2;
4 Output the best packing $\mathcal{P}$ among $\mathcal{P}_L$ and $\mathcal{P}_R$.
```

The time complexity of this algorithm is mainly the time complexity of computing a maximum weight matching in graphs of maximum degree 9, that is $O(|V|^2 \log |V|)$, [14].

**Theorem 4.3** *`Weighted` $\mathbf{P}_3$-`Packing` provides a 1/2-approximation for* MaxW$\mathbf{P}_3$Packing *in bipartite graphs with maximum degree 3 and this ratio is tight.*

**Proof:** Let $\mathcal{P}^*$ be an optimum $P_3$-packing on $I = (G, w)$, we denote by $\mathcal{P}_L^*$ (resp., $\mathcal{P}_R^*$) the paths of $\mathcal{P}^*$ of which the two endpoints belong to $L$ (resp., $R$); thus, $opt(I) = w(\mathcal{P}_L^*) + w(\mathcal{P}_L^*)$. For any path $P = P_{x,y} \in \mathcal{P}_L^*$, $[x, y]$ is an edge from $E_L$, of weight $d_L(x, y) \geq w(P_{x,y})$. Hence, $M_L = \{[x, y] | P_{x,y} \in \mathcal{P}_L^*\}$ is a matching on $G_L$ that verifies:

$$d(M_L) \geq w(\mathcal{P}_L^*) \tag{1}$$

Moreover, since $M_L^*$ is a maximum weight matching on $G_L$, we have $d_L(M_L) \leq d_L(M_L^*)$. Thus, using inequality (1) and Lemma 4.2 (and by applying the same arguments on $G_R$), we deduce:

$$w(\mathcal{P}_L) \geq w(\mathcal{P}_L^*),\ w(\mathcal{P}_R) \geq w(\mathcal{P}_R^*) \tag{2}$$

Finally, the solution outputted by the algorithm verifies $w(\mathcal{P}) \geq 1/2(w(\mathcal{P}_L) + w(\mathcal{P}_R))$; thus, we directly deduce from inequalities (2) the expected result. The instance $I = (G, w)$ that provides the tightness is depicted in Figure 6. It consists of a graph on $12n$ vertices on which one can easily observe that $w(\mathcal{P}_L) = w(\mathcal{P}_R) = 2n(n + 2)$ and $w(\mathcal{P}^*) = 2n(2n + 2)$. ■

Concerning the unweighted case, we may obtain the same performance ratio without the restriction on the maximum degree of the graph. Concerning the unweighted case, we may obtain the same performance ratio without the restriction on the maximum degree of the graph. **The main differences with the previous algorithm lies on the construction of the two graphs $G_L$, $G_R$:** starting from $G$, we duplicate each vertex $r_i \in R$ by adding a new vertex $r_i'$ with the same neighborhood than $r_i$ (this operation, often called *multiplication of vertices* in the literature, is notably used in the characterization of perfect graphs); finally, we add the edge $[r_i, r'_i]$. If $R_L$ denotes the vertex set $\{r_i, r'_i | r_i \in R\}$, then the following property holds:

**Property 4.4** *From any matching $M$ on $G_L$, one can deduce a matching $M'$ on $G_L$ that saturates $R_L$, and such that $|M'| \geq |M|$.*

Let $M$ be a matching on $G_L$. If none of the two vertices $r_i$ and $r'_i$ for some $i$ are saturated by $M$, then set $M' = M \cup \{[r_i, r'_i]\}$. If exactly one of them is saturated by a given edge $e$ from $M$, then set $M' = (M \setminus \{e\}) \cup \{[r_i, r'_i]\}$. In any case, $M'$ is still a matching of size at least $|M|$. Thus, the expected result is obtained by applying this process to each vertex of $R_L$.

**Theorem 4.5** *There is a $1/2$-approximation for* MaxP3Packing *in bipartite graphs and this ratio is tight. The complexity time of this algorithm is $O(m\sqrt{n})$.*

**Proof:** See appendix. ■

## 4.3 Min3-PathPartition in general graphs

To our knowledge, the approximability of Min$k$-PathPartition (or MinPathPartition) has not been studied so far. Here, we propose a 3/2-approximation for Min3-PathPartition. Note that, concerning MinPathPartition (that is, the approximation of $\rho(G)$), we can trivially see that it is not $(2-\varepsilon)$-approximable, from the fact that deciding whether $\rho(G) = 1$ or $\rho(G) \geq 2$ is **NP**-complete. Actually, we can more generally establish that $\rho(G)$ is not in **APX**: otherwise, we could obtain a **PTAS** for the traveling salesman problem with weight 1 and 2 when $opt(I) = n$, which is not possible, unless **P**=**NP**.

`Computing` $\rho_2(G)$

```
1 Compute a maximum matching $M_1^*$ on $G$;

2 Build a bipartite graph $G_2 = (L, R; E_2)$ where $L = \{l_e | e \in M_1^*\}$, $R = \{r_v | v \in V \setminus V(M_1^*)\}$, and $[l_e, r_v] \in E_2$ iff the corresponding isolated vertex $v \notin V(M_1^*)$ is adjacent in $G$ to the edge $e \in M_1^*$;

3 Compute a maximum matching $M_2^*$ on $G_2$;

4 Output $\mathcal{P}'$ the 3-paths partition deduced from $M_1^*$, $M_2^*$, and $V \setminus V(M_1^* \cup M_2^*)$. Precisely, if $M'_1 \subseteq M_1^*$ is the set of edges adjacent to $M_2^*$, then the paths of length 2 are given by $M'_1 \cup M_2^*$, the paths of length 1 are given by $M_1^* \setminus M'_1$, and the paths of length 0 (that is, the isolated vertices) are given by $V \setminus V(M_1^* \cup M_2^*)$;
```

The time complexity of this algorithm is $O(nm + n^2 \log n)$, [14].

**Theorem 4.6** Min3-PathPartition *is 3/2-approximable in general graphs; this ratio is tight for the algorithm we analyse.*

**Proof:** Let $G = (V, E)$ be an instance of Min3-PathPartition. Let $\mathcal{P}^* = (\mathcal{P}_2^*, \mathcal{P}_1^*, \mathcal{P}_0^*)$ and $\mathcal{P}' = (\mathcal{P}'_2, \mathcal{P}'_1, \mathcal{P}'_0)$ respectively be an optimal solution and the approximate 3-path partition on $G$, where $\mathcal{P}_i^*$ and $\mathcal{P}_i'$ denote for $i = 0, 1, 2$ the set of paths of length $i$. By construction of the approximate solution, we have:

$$apx(I) = |V| - 2|M_1^*| - |M_2^*| \tag{3}$$

Let $V_0 = (V \backslash V(M_1^*)) \backslash \mathcal{P}_0^*$, we consider a subgraph $G_2' = (L, R'; E_2')$ of $G_2$, where $R'$ and $E_2'$ are defined as: $R' = \{r_v \in R | v \in V_0\}$ and $E_2'$ contains the edge $[l_e, r_v] \in E_2'$ *iff* there is an edge of $\mathcal{P}^*$ that links $v$ to an endpoint of $e$. By construction of $V_0$, we deduce that $d_{G_2'}(r_v) \geq 1$ for any $v \in V_0$ (consider that $V_0$ is an independent set of $G$, because of $M_1^*$ optimality). Moreover, we have $d_{G_2'}(l_e) \leq 2$ for any $e \in M_1^*$. Assume the reverse: then we can delete $e$ and add two edges of $\mathcal{P}^*$ in $M_1^*$ (consider that $\mathcal{P}^*$ is a packing), which contradicts once again the optimality of $M_1^*$. Thus, we deduce that $G_2'$ contains a matching that is of size at least one-half $|R'|$; hence the same holds for $G_2$ and we get:

$$|M_2^*| \geq 1/2|R'| \geq 1/2\,(|V| - 2|M_1^*| - |\mathcal{P}_0^*|) \tag{4}$$

From relations (3) and (4), we deduce:

$$\begin{aligned} apx(I) &= |V| - 2|M_1^*| - |M_2^*| \\ &\leq |V| - 2|M_1^*| - 1/2\,(|V| - 2|M_1^*| - |\mathcal{P}_0^*|) \\ &\leq 1/2\,(|V| + |\mathcal{P}_0^*|) - |M_1^*| \end{aligned}$$

Now, consider the optimal solution. From $|V| = 3|\mathcal{P}_2^*| + 2|\mathcal{P}_1^*| + |\mathcal{P}_0^*|$, we trivially have:

$$\begin{aligned} opt(I) &= |\mathcal{P}_2^*| + |\mathcal{P}_1^*| + |\mathcal{P}_0^*| \\ &= 1/3\,(3|\mathcal{P}_2^*| + 3|\mathcal{P}_1^*| + 3|\mathcal{P}_0^*|) \\ &\geq 1/3\,(|V| + |\mathcal{P}_0^*|) \end{aligned}$$

Thus, we obtain the expected result. The proof of the tightness is omitted.

■

## References

[1] E. Arkin, R. Hassin. On local search for weighted packing problems. *Mathematics of Operations Research*, 23: 640-648, 1998.

[2] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela and M. Protasi. Complexity and Approximation (Combinatorial Optimization Problems and Their Approximability Properties). *Springer, Berlin*, 1999.

[3] C. Bazgan, R. Hassin, and J. Monnot. Approximation algorithms for some routing problems. *Discrete Applied Mathematics*, 146: 3-26, 2005.

[4] B. Chazelle. A minimum spanning tree algorithm with Inverse-Ackermann type complexity. *J. ACM*, 47: 1028-1047, 2000.

[5] B. Csaba, M. Karpinski, P. Krysta. Approximability of dense and sparse instances of minimum 2-connectivity, TSP and path problems. *SODA*, 74-83, 2002.

[6] M. Dyer, A. Frieze. Planar 3DM is NP-complete. *J. Algorithms*, 7:174-184, 1986.

[7] A. Frank. Some Polynomial Time Algorithms for Certain Graphs and Hypergraphs. *Proceedings of the 5th British Combinatorial Conference, Congressus Numerantium XV, Utilitas Mathematicae, Winnipeg*, 211-226, 1976).

[8] M. R. Garey, D. S. Johnson. Computers and intractability. A guide to the theory of NP-completeness. *CA, Freeman*, 1979.

[9] R. Hassin, S. Rubinstein. An Approximation Algorithm for Maximum Packing of 3-Edge Paths. *Inf. Process. Lett.*, 63: 63-67, 1997.

[10] R. Hassin, S. Rubinstein. An Approximation Algorithm for Maximum Triangle Packing. *ESA* , LNCS 3221: 403-413, 2004.

[11] A. Kaneko. A necessary and sufficient condition for the existence of a path factor every component of which is a path of length at least two. *Journal of Combinatorial Theory, Series B*, 88: 195-218, 2003.

[12] M. Karpinski. Personnal communication. 2006.

[13] D. G. Kirkpatrick, P. Hell. On the Completeness of a Generalized Matching Problem. *Proc. STOC'78*, 240-245, 1978.

[14] L. Lovasz, M. D. Plummer. Matching Theory. *North-Holland,*, Amsterdam, 1986.

[15] J. Monnot, S. Toulouse. Approximation results for the weighted $P_4$ partition problem. *The symposia on Fundamentals of Computation Theory*, F.C.T.'2005, LNCS 3623, 377-385, 2005.

[16] E. Petrank. The Hardness of Approximation: Gap Location. *Computational Complexity*, 4, 133-157, 1994.

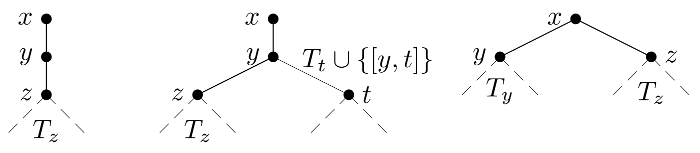
[17] G. Steiner. On the k-Path partition problem in cographs. *Cong. Numer.*, 147:89-96, 2000.

[18] G. Steiner. k-Path partitions in trees. *Theor. Comput. Sci.*, 290:2147-2155, 2003.

[19] H. Wang. Path factors of bipartite graphs. *Journal of Graph Theory*, 18: 161-167, 1994.

[20] S. Vishwanathan. An Approximation Algorithm for the Asymmetric Travelling Salesman Problem with Distances One and Two. *Information Processing Letter*, 44(6): 297-302, 1992.

[21] J-H Yan, G. J. Chang, S. M. Hedetniemi, S.T. Hedetniemi. On the k-path partition of graphs. *Discrete Applied Mathematics*, 78:227-233.



Steps 3.1 and 3.2 Steps 3.3 and 3.4 Steps from 4 to 4.3

Figure 7: The main configurations of the algorithm.

## Appendix

**Proof of Proposition 3.3**. We reduce the problem of computing an optimum solution of MaxW$\mathbf{P}_k$Packing in graphs with maximum degree 2 (or in a forest) to the problem of computing a maximum weight independent set (MaxWIS in short) in an interval (or chordal) graph, which is known to be polynomial, [7].

Let $I = (G, w)$ be an instance of MaxW$\mathbf{P}_k$Packing where $G = (V, E)$ is a graph with maximum degree 2. Hence, $G$ is a collection of disjoint paths or cycles and thus, each connected component may be separately solved. Moreover, wlog., we may assume that each connected component $G^\ell$ of $G$ is a path. Otherwise, a given cycle $G^\ell = \{v_1, \ldots, v_{N_\ell}, v_1\}$ might be solved by picking the best solution among the solutions computed on the $k$ instances $G^\ell \setminus \{[v_1, v_2]\}, \ldots, G^\ell \setminus \{[v_k, v_{k+1}]\}$.

Thus, let $G^\ell = \left\{v_1^\ell, \ldots, v_{N^\ell}^\ell\right\}$ be such a path; we build the instance $(H^\ell, w^\ell)$ of MaxWIS where the vertex set of $H^\ell$ corresponds to the paths of length $k-1$ in $G^\ell$: a vertex $v$ corresponding to a path $P_v$ has a weight $w^\ell(v) = w(P_v)$. Moreover, two vertices $u \neq v$ are linked in $H^\ell$ *iff* the corresponding paths $P_u$ and $P_v$ share at least one common vertex in the initial graph. We deduce that the set of independent sets in $H^\ell$ corresponds to the set of $\mathbf{P}_k$ in $G^\ell$. Observe that $H^\ell$ is an interval graph (even a unit interval graph), since each path can be viewed as an interval of the line $\{1, \cdots, N^\ell\}$. Hence, $H^\ell$ is chordal.

If $G$ is a forest, then any of the graphs $H^\ell$ that correspond to a tree of $G$ is a chordal graph.

**Proof of Theorem 4.1**. The approximate algorithm is based on the following observation: for any spanning tree $T = (V_T, E_T)$ of maximum degree 3 and containing at least 3 vertices, we can build a coverage of its edge set into 3 packings of $\mathbf{P}_3$ within linear time. Consider three empty collections $\mathcal{P}^1, \mathcal{P}^2, \mathcal{P}^3$ and a tree rooted at $r$. According to the degree of $r$ and to the degree of its children, we add some path $P$ using $r$ to the packing $\mathcal{P}^1$, remove the edges of $P$ from $T$, and then recursively repeat this process on the remaining subtrees rooted at its children, alternatively invoking $\mathcal{P}^2$ and $\mathcal{P}^1$. Figure 7 provides an illustration of this process, where $T_v$ denotes the subtree of $T$ rooted at $v$; the edges in rigid lines represent the path that is added to the current packing; finally, the subtrees that are invoked by the recursive calls are indicated.

We start the whole process by picking a vertex $r$ of degree at most 2 as a root of the initial tree. The stopping criterion are the following: the tree has no edge (then stop), or the tree is a lonely edge $[x, y]$; then add $\{r_x, x, y\}$ to $\mathcal{P}^3$, where $r_x$ denotes the father of $x$ in $T$. A formal description of this algorithm is provided below.

```
Tree-P_3PackingCover
    Input: $T = (V_T, E_T)$ spanning tree of maximum degree 3 containing at least 3 vertices
    and rooted at $r$ such that $d_T(r) \leq 2$.
  1 Set $\mathcal{P}^1 = \mathcal{P}^2 = \mathcal{P}^3 = \emptyset$;
  2 Call SubProcess($T_r$,$\mathcal{P}^1$,$\mathcal{P}^2$,$\mathcal{P}^3$,1);
  3 Repair $\mathcal{P}^1, \mathcal{P}^2, \mathcal{P}^3$ in such a way that each $\mathcal{P}^i$ is a packing;
    Output ($\mathcal{P}^1$, $\mathcal{P}^2$, $\mathcal{P}^3$).

SubProcess($T_x$, $\mathcal{P}^1$,$\mathcal{P}^2$,$\mathcal{P}^3$, $i$)
  1 If $E_{T_x} = \emptyset$ then exit;
  2 If $E_{T_x} = \{\{x, y\}\}$
2.1     Let $r_x$ be the father of $x$ in $T_r$, set $\mathcal{P}^3 = \mathcal{P}^3 \cup \{\{r_x, x, y\}\}$;
  3 Else If $x$ is of degree 1 in $T_x$ with child $y$
3.1     Let $z$ be a child of $y$, set $\mathcal{P}^i = \mathcal{P}^i \cup \{\{x, y, z\}\}$;
3.2     Call SubProcess($T_z$, $\mathcal{P}^1$,$\mathcal{P}^2$,$\mathcal{P}^3$,3-$i$);
3.3     If $y$ is of degree 3 in $T_x$, let $t$ denote its second child,
            Call SubProcess($\{\{y, t\}\} \cup T_t$, $\mathcal{P}^1$,$\mathcal{P}^2$,$\mathcal{P}^3$, 3-$i$);
  4 Else If $x$ is of degree 2 in $T_x$ with children $y$ and $z$
4.1     Set $\mathcal{P}^i = \mathcal{P}^i \cup \{\{y, x, z\}\}$;
4.2     Call SubProcess($T_y$, $\mathcal{P}^1$,$\mathcal{P}^2$,$\mathcal{P}^3$,3-$i$);
4.3     Call SubProcess($T_z$, $\mathcal{P}^1$,$\mathcal{P}^2$,$\mathcal{P}^3$,3-$i$);
```

Doing so, $\mathcal{P}^1$ and $\mathcal{P}^2$ both are packings: one can easily see that the paths that are added to $\mathcal{P}^i$ (where $i = 1$ or $i = 2$) at a given time $t$ and the ones that are added again to $\mathcal{P}^i$ at time $t + 2$ do not share common vertices. On the other hand, $\mathcal{P}^3$ might not be a packing. Let $\{r_x, x, y\}$ and $\{r_{x'}, x', y'\}$ be two paths from $\mathcal{P}^3$ such that $\{r_x, x, y\} \cap \{r_{x'}, x', y'\} \neq \emptyset$; then, either $r_x = r_{x'}$, or $r_x = x'$. If the first case occurs, $\{x, r_x, x'\}$ has been added to $\mathcal{P}^i$ (for $i = 1$ or $i = 2$), then set: $\mathcal{P}^i = \mathcal{P}^i \backslash \{\{x, r_x, x'\}\} \cup \{\{r_x, x, y\}\}$ and $\mathcal{P}^3 = \mathcal{P}^3 \backslash \{\{r_x, x, y\}\}$. Otherwise, $r_{x'}$ is the father of $r_x$ in $T_r$ and we have $\{r_{x'}, r_x, x\} \in \mathcal{P}^i$ (for $i = 1$ or $i = 2$); then set: $\mathcal{P}^i = \mathcal{P}^i \backslash \{\{r_{x'}, r_x, x\}\} \cup \{\{r_{x'}, x', y'\}\}$ and $\mathcal{P}^3 = \mathcal{P}^3 \backslash \{\{r_{x'}, x', y'\}\}$. Figure 8 provides 2 examples of the construction of $\mathcal{P}^1$, $\mathcal{P}^2$ and $\mathcal{P}^3$.

Hence, the following algorithm provides a 1/3-approximation within $O(\alpha(n, 3n/2)n)$ time complexity.

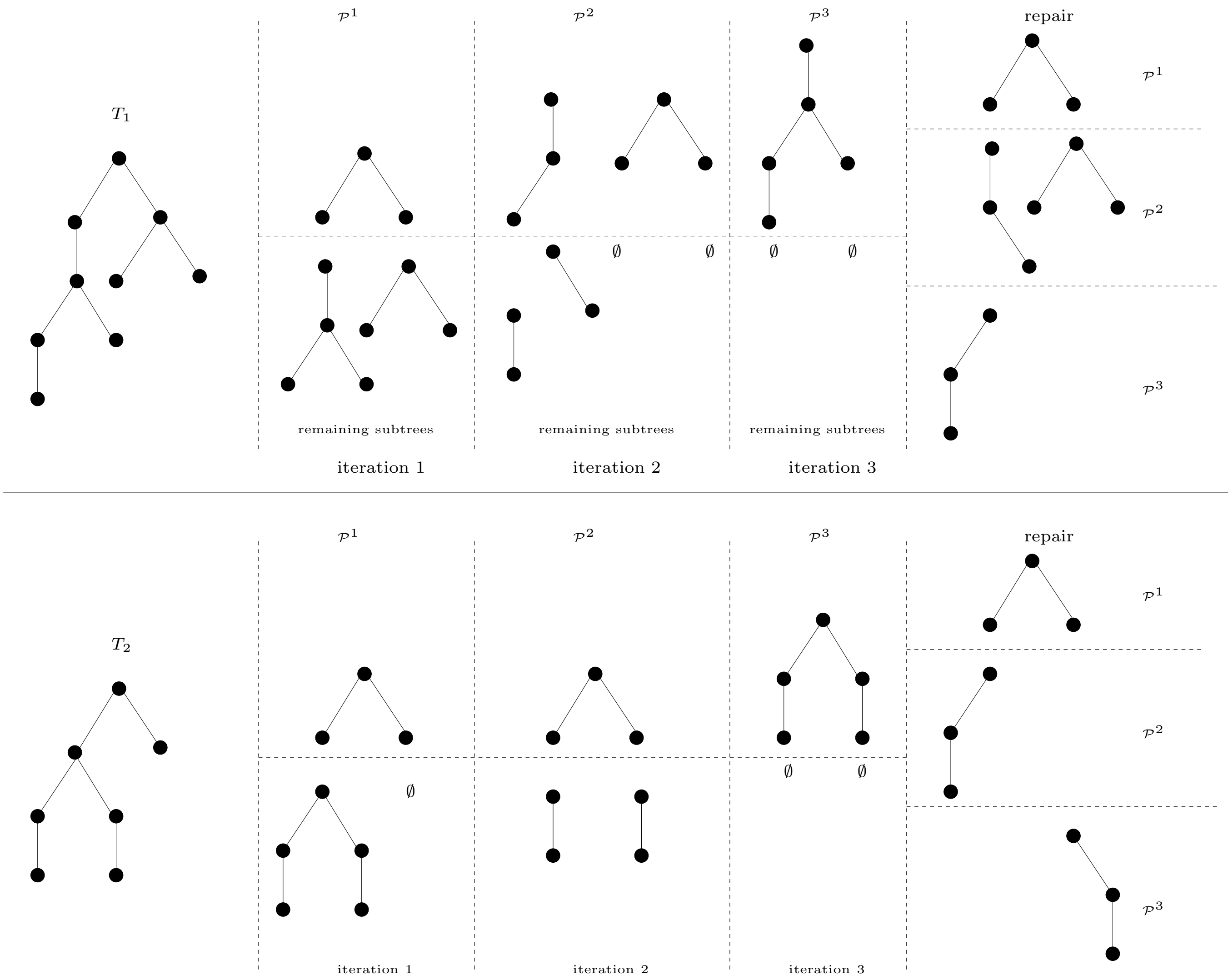


Figure 8: Two examples of the construction of the 3 packings $\mathcal{P}^i$ for $i = 1, 2, 3$.

```
MaxW$\mathbf{P}_3$Packing
    Input: $G = (V, E)$ weighted connected graph with maximum degree 3.
1   Compute a maximum-weight spanning tree $T$ on $G$;
2   Compute $(\mathcal{P}^1, \mathcal{P}^2, \mathcal{P}^3)$ a $\mathbf{P}_3$-packing cover of $T$;
    Output $\mathcal{P} = \arg\max\{w(\mathcal{P}^1), w(\mathcal{P}^2), w(\mathcal{P}^3)\}$;
```

First, the computation of a maximum weight spanning tree can be done in $O(\alpha(n, 3n/2)n)$ time in graphs with maximum degree 3, [4]; second, the number of recursive calls to `SubProcess` may not exceed $2/3n$ and finally, $|\mathcal{P}^3|$ is at most $O(\log n)$.

**Proof of Theorem 4.5**. The approximate algorithm works as previously, except that we compute a maximum (size) matching $M_L^*$ (resp., $M_R^*$) on $G_L$ (resp., $G_R$) in step 2 and that the $\mathbf{P}_3$ packing $\mathcal{P}_L$ (resp., $\mathcal{P}_R$) is obtained from $M_L^*$ (resp., $M_R^*$) by deleting the edges $[r_i, r_i']$ (resp., $[l_i, l_i']$) in step 3.

Let $M$ be a matching on $G_L$; from the property 4.4, we may assume that $M$ saturates $R_L$. Hence, if $p$ is the number of edges $[r_i, r_i']$ of $M$, we have $|M| = |R_L| - p = 2|R| - p$. We deduce from $M$ a packing $\mathcal{P}_M$ of 2-length-paths in $G$, using $|M| - p = 2(|M| - |R|)$ edges. Conversely, any packing $\mathcal{P}_L = \{P_1, \cdots, P_q\}$ on $G$ such that for $1 \leq i \leq q$, $P_i$ is a 2-length-path with its two endpoints in $L$ can be converted into a matching $M$ on $G_L$ of size $|M| = 2q + (|R| - q) = q + |R|$. Thus, from a maximum matching $M^*$ on $G_L$ verifying Property 4.4, we build a $\mathbf{P_3}$ packing $\mathcal{P}_{M^*}$ on $G$ using $2(|M^*| - |R|)$ edges. Since $2(|M^*| - |R|) \geq 2|\mathcal{P}_L^*|$, the proof is complete. The complexity time of this algorithm is mainly the complexity time of computing a maximum matching, that is $O(m\sqrt{n})$, [14].