# On the complexity of the exact weighted independent set problem

*Martin Milanič*[*] *Jérôme Monnot*[†‡]

**Abstract**

We introduce and study the exact weighted independent set problem: Given a vertex-weighted graph, determine whether it contains an independent set of a given weight. This problem is related to the exact perfect matching problem. We determine the complexity of the problem and the variant of it in which the independent set is required to be a maximum one, for several graph classes. More specifically, we show that these problems are strongly **NP**-complete for several subclasses of bipartite graphs, including the class of bipartite graphs of maximum degree 3. On the positive side, we identify graph classes where these problems are solvable in pseudo-polynomial time. We complement the dynamic programming solutions by showing that modular decomposition can be applied to a suitable generalization of these "exact" problems.

**Keywords:** exact weighted independent set, **NP**-completeness, pseudo-polynomial algorithm, modular decomposition.

## 1 Introduction

The exact perfect matching problem is the problem of determining whether a given edge-weighted graph contains a perfect matching of a given weight. This problem finds applications in such diverse areas as bus-driver scheduling, statistical mechanics [26], DNA sequencing [7], and robust assignment problems [17]. In 1982, Papadimitriou and Yannakakis showed that the problem—as well as many other *exact* versions of polynomially solvable optimization problems—is **NP**-complete when the weights are encoded in binary [32]. The problem is solvable in polynomial time in some special cases such as planar graphs (and more generally, for graphs that have a Pfaffian orientation, provided one is given) [5], and complete or complete bipartite graphs with 0-1 weights [25], and admits a randomized pseudo-polynomial-time algorithm [30]. However, the deterministic complexity of the exact perfect matching problem with unary weights remains unsettled, even for bipartite graphs. Papadimitriou and Yannakakis conjectured that the problem is **NP**-complete [32].

Motivated by this long-standing open problem, we introduce and study the *exact weighted independent set* problem and a restricted version of it, both closely related to the exact perfect matching problem. An *independent set* (also called *stable set*) in a graph is a set of pairwise non-adjacent vertices. The *weighted independent set* problem (WIS) asks for an independent

[*]FAMNIT and PINT, University of Primorska, Koper, Slovenia, martin.milanic@upr.si

[†]LAMSADE, Université Paris-Dauphine, Place du Maréchal De Lattre de Tassigny, F-75775 Paris Cedex 16, France, monnot@lamsade.dauphine.fr

[‡]Part of this work was carried out while the first author was with the LAMSADE on a visiting researcher position supported by a common CNRS-NSF research project "Algorithmic Decision Theory" between the LAMSADE and the DIMACS. Financial support of DIMACS, as well as the hospitality of members of LAMSADE are gratefully acknowledged.

set of maximum weight in a given vertex-weighted graph $(G, w)$. If all weights are the same, we speak about the *independent set* problem (IS), which consists in finding an independent set of maximum cardinality. The optimal values of these problems are denoted by $\alpha_w(G)$ and $\alpha(G)$, respectively.

The *exact weighted independent set* problem (EWIS) consists of determining whether a given vertex-weighted graph $(G, w)$ with $G = (V, E)$ and $w : V \to \mathbb{Z}$ contains an independent set whose total weight (that is, the sum of the weights of its members) equals a given integer $M$. The *exact weighted maximum independent set* problem ($\text{EWIS}_\alpha$) is the restriction of the EWIS problem where we require the independent set to be one of the maximum cardinality. Thus, given a vertex-weighted graph $(G, w)$ and an integer $M$, $\text{EWIS}_\alpha$ asks about the existence of an independent set $I$ of $G$ with $|I| = \alpha(G)$ and $w(I) = M$.

The connection between the exact perfect matching problem and the exact weighted independent set problem can be explained through line graphs. The *line graph* $L(G)$ of a graph $G = (V, E)$ is the graph whose vertex set is $E$, and whose two vertices are adjacent if and only if they share a common vertex as edges of $G$. Clearly, there is a one-to-one correspondence between the matchings of a graph and the independent sets of its line graph. Thus, the exact matching problem is precisely the exact weighted independent set problem, restricted to the class of line graphs. Similarly, under the (polynomially verifiable) assumption that the input graph has a perfect matching, the exact perfect matching problem is precisely the exact weighted maximum independent set problem, restricted to the class of the line graphs of graphs with a perfect matching.

**Our contributions.** We determine the complexity status of EWIS and $\text{EWIS}_\alpha$ for several graph classes. On the one hand, we present the first nontrivial strong **NP**-completeness result for these problems. On the other hand, we identify several classes of graphs where EWIS and $\text{EWIS}_\alpha$ can be solved in pseudo-polynomial time. While most of the solutions are based on a dynamic programming approach, we also show that modular decomposition can be applied to a suitable generalization of the EWIS problem.

The paper is structured as follows. After giving the necessary definitions and notations, we continue the introductory discussion in Section 2. We also present some relations between the complexities of the problems WIS, EWIS and $\text{EWIS}_\alpha$. Section 3 is devoted to the strong **NP**-completeness results. In Section 4, pseudo-polynomial-time solutions to the exact weighted independent set problem are discussed. We conclude the paper with a short discussion in Section 5.

**Basic definitions and notations.** All graphs considered are finite, simple and undirected. A *matching* in a graph is a set of pairwise disjoint edges. A matching is *perfect* if every vertex of the graph is contained in some edge from the matching. Unless otherwise stated, $n$ and $m$ will denote the number of vertices and edges, respectively, of the graph considered. As usual, $P_n$ and $C_n$ denote the chordless path and the chordless cycle on $n$ vertices. By $K_n$ we denote the complete graph on $n$ vertices, and by $K_{s,t}$ the complete bipartite graph with parts of size $s$ and $t$. A *claw* is the graph $K_{1,3}$. A *net* is the graph obtained from a triangle by attaching one pendant edge to each vertex.

For a graph $G$, we will denote by $V(G)$ and $E(G)$ the vertex-set and the edge-set of $G$, respectively. Individual edges will be denoted by square brackets: an edge with endpoints $u$ and $v$ will be denoted by $[u, v]$. For a vertex $x$ in a graph $G$, we denote by $N_G(x)$ the neighborhood of $x$ in $G$, i.e., the set of vertices adjacent to $x$, and by $N_G[x]$ the closed neighborhood of $x$, i.e., the set $N_G(x) \cup \{x\}$. We will write $N(x)$ and $N[x]$ instead of $N_G(x)$ and $N_G[x]$ if no confusion can arise. For a graph $G$, we denote by co-$G$ (also $\overline{G}$) the edge-complement of $G$. By component we will always mean a connected component. For graph-theoretical terms not

defined here, the reader is referred to Berge's book [6].

The triple $(G, w, M)$ will always represent an instance of EWIS (or EWIS$_\alpha$), i.e., $G = (V, E)$ is a graph, $w : V \to \mathbb{Z}$ are vertex weights, and $M \in \mathbb{Z}$ is the target weight. If $H$ is an induced subgraph of $G$, we will also consider triples of the form $(H, w, M)$ as instances of EWIS, with $w$ representing the restriction of the weights to $V(H)$. We will denote by EWIS$(G, w, M)$ the solution to the instance $(G, w, M)$ of EWIS, that is, EWIS$(G, w, M)$ is *yes* if there is an independent set $I$ in $G$ with $w(I) = M$, and *no* otherwise. Similarly, EWIS$_\alpha(G, w, M)$ is *yes* if there is a maximum independent set $I$ in $G$ with $w(I) = M$, and *no* otherwise. Finally, for a subset of vertices $V' \subseteq V$, we let $w(V') = \sum_{v \in V'} w(v)$. For a positive integer $k$, we write $[k]$ for the set $\{1, \ldots, k\}$.

## 2 Preliminary Observations

The exact weighted independent set problem is (weakly) **NP**-complete for any class of graphs containing $\{\overline{K_n} : n \geq 0\}$. There is a direct equivalence between the exact weighted independent set problem on $\{nK_1 : n \geq 0\}$ and the **NP**-complete *subset sum* problem [20]: Given $n$ integers $a_1, \ldots, a_n$ and an integer $b$, determine whether there is a subset $J \subseteq [n]$ such that $\sum_{j \in J} a_j = b$. Therefore, for a given class of graphs $\mathcal{G}$, the question of interest is whether the EWIS problem is strongly **NP**-complete for graphs in $\mathcal{G}$, or is it solvable in pseudo-polynomial time.

By the use of simple reductions, it is easy to see that it suffices to consider instances $(G, w, M)$ of EWIS such that $1 \leq w(v) \leq M$, for all $v \in V(G)$, as well as $M \leq w(V(G))$. The same assumption can also be made for the instances $(G, w, M)$ of the restricted counterpart EWIS$_\alpha$.

We now discuss some relations between the complexities of the problems WIS, EWIS and EWIS$_\alpha$, when restricted to particular graph classes.

**Lemma 1.** *Let $\mathcal{G}$ be a class of graphs. The following statements are true.*
*(i) If* EWIS$_\alpha$ *is solvable in pseudo-polynomial time for graphs in $\mathcal{G}$, then* WIS *is solvable in pseudo-polynomial time for graphs in $\mathcal{G}$.*
*(ii) If* EWIS *is solvable in pseudo-polynomial time for graphs in $\mathcal{G}$, then* EWIS$_\alpha$ *is solvable in pseudo-polynomial time for graphs in $\mathcal{G}$.*
*(iii) Let $\mathcal{G}' = \{G' : G \in \mathcal{G}\}$ where $G' = (V', E')$ is the graph, obtained from a graph $G = (V, E) \in \mathcal{G}$, by adding pendant vertices, as follows: $V' = V \cup \{v' : v \in V\}$, $E' = E \cup \{[v, v'] : v \in V\}$. If* EWIS$_\alpha$ *is solvable in pseudo-polynomial time for graphs in $\mathcal{G}'$, then* EWIS *is solvable in pseudo-polynomial time for graphs in $\mathcal{G}$.*

*Proof.* $(i)$ Let $(G, w, k)$ be an instance of the decision version of the weighted independent set problem. As we can assume positive weights, $G$ contains an independent set of total weight at least $k$ if and only if $G$ contains a maximum independent set of total weight at least $k$. By testing values for $M$ from $w(V)$ down to $k$ and using an algorithm for EWIS$_\alpha$ on the instance $(G, w, M)$, we can decide whether $G$ contains a maximum independent set of total weight at least $k$.
$(ii)$ Let $(G, w, M)$ be an instance of the EWIS$_\alpha$ problem. It is easy to see that the following algorithm solves EWIS$_\alpha$.
*Step 1.* Compute $\alpha(G)$, which is equal to the maximum $k \in [n]$ such that the solution to EWIS$(G, \mathbf{1}, k)$ is *yes*, where $\mathbf{1}$ denotes the unit vertex weights.
*Step 2.* Let $N = w(V) + 1$. For every vertex $v \in V(G)$, let $w'(v) = w(v) + N$. Let $M' = M + \alpha(G)N$. Then it is easy to verify that EWIS$_\alpha(G, w, M) =$ EWIS$(G, w', M')$.

(*iii*) Let $(G, w, M)$ with $G = (V, E) \in \mathcal{G}$ be an instance of EWIS. Let $G'$ be the graph, defined as in the lemma. Let $n = |V|$ and let $w'(v) = (n+1)w(v)$ for all $v \in V$ and $w'(v) = 1$ for $v \in V'$. Then, it is easy to verify that the solution to EWIS$(G, w, M)$ is *yes* if and only if the solution to EWIS$_\alpha(G', w', M')$ is *yes* for some value $M'$ in the set $\{(n+1)M, \ldots, (n+1)M+n-1\}$. □

The EWIS problem is clearly in **NP**, and so is EWIS$_\alpha$ for any class of graphs $\mathcal{G}$ where IS is polynomially solvable. Therefore, Lemma 1 implies the following result.

**Corollary 2.** *Let $\mathcal{G}$ be a class of graphs. The following statements are true.*
*(i) If* WIS *is strongly* ***NP****-complete for graphs in $\mathcal{G}$, then* EWIS$_\alpha$ *is strongly* ***NP****-hard for graphs in $\mathcal{G}$. If, in addition,* IS *is polynomial for graphs in $\mathcal{G}$, then* EWIS$_\alpha$ *is strongly* ***NP****-complete for graphs in $\mathcal{G}$.*
*(ii) If* EWIS$_\alpha$ *is strongly* ***NP****-hard for graphs in $\mathcal{G}$, then* EWIS *is strongly* ***NP****-complete for graphs in $\mathcal{G}$.*
*(iii) Let $\mathcal{G}'$ be as in Lemma 1. If* EWIS *is strongly* ***NP****-complete for graphs in $\mathcal{G}$, then* EWIS$_\alpha$ *is strongly* ***NP****-hard for graphs in $\mathcal{G}'$. If, in addition,* IS *is polynomial for graphs in $\mathcal{G}'$, then* EWIS$_\alpha$ *is strongly* ***NP****-complete for graphs in $\mathcal{G}'$.*

We can thus safely focus in determining the complexity of the EWIS problem for those graph classes where the WIS problem is solvable in pseudo-polynomial time. Moreover, combining parts (*ii*) and (*iii*) of the above corollary shows that when $\mathcal{G} \in \{$forests, bipartite graphs, chordal graphs$\}$, the problems EWIS and EWIS$_\alpha$ are equivalent (in the sense that, when restricted to the graphs in $\mathcal{G}$, they are either both solvable in pseudo-polynomial time, or they are both strongly **NP**-complete). Recall that a graph $G$ is a *forest* if it is acyclic, *bipartite* if any cycle of $G$ has even length, and *chordal* if any cycle of $G$ with size at least 4 has a chord (i.e., an edge connecting two non-consecutive vertices of the cycle).

We conclude this section by showing that a similar equivalence remains valid for the class of line graphs. More precisely, if $L$, $L(Bip)$, $L(K_{2n})$ and $L(K_{n,n})$ denote the classes of line graphs, line graphs of bipartite graphs, line graphs of complete graphs with an even number of vertices, and line graphs of complete balanced bipartite graphs, respectively, we have the following result.

**Lemma 3.** EWIS *is strongly* ***NP****-complete for graphs in $L$ (resp., $L(Bip)$) if and only if* EWIS$_\alpha$ *is strongly* ***NP****-complete for graphs in $L(K_{2n})$ (resp., $L(K_{n,n})$).*

*Proof.* The backward implication is given by part (*ii*) of Lemma 1. The forward implication follows from a reduction of the exact matching problem to the exact perfect matching problem which we show now. Given an instance $G = (V, E)$ with edge weights $w$ and a target $M$ for the exact matching problem, construct an instance $(K_{n'}, w', M')$ for the exact perfect matching problem as follows. If $n = |V|$ is odd, we add a new vertex and we complete the graph $G$. For an edge $e$ of $G$, let $w'(e) = Nw(e)$ where $N = w(E) + 1$, for an edge $e \notin E$ let $w'(e) = 1$. The transformation is clearly polynomial, and $G$ has a matching of weight $M$ if and only if $K_{n'}$ has a perfect matching of weight $NM + k$ for some value of $k \in \{0, \ldots, n-1\}$. Also, it is easy to see that in the case of bipartite graphs $G = (L, R; E)$ with $|L| \leq |R|$, we can add $|R \setminus L|$ vertices to $L$ to balance the bipartition. □

## 3 Hardness Results

The weighted independent set problem is solvable in polynomial time for bipartite graphs by network flow techniques. However, as we show in this section, the exact versions of the problem turn out to be strongly **NP**-complete for several subclasses of bipartite graphs, including the

class of bipartite graphs of maximum degree 3. By part $(ii)$ of Lemma 1, it suffices to show the hardness of the restricted version EWIS$_\alpha$.

### 3.1 EWIS$_\alpha$ in Bipartite Graphs

A bipartite graph is a graph $G = (V, E)$ whose vertex set admits a partition $V = L \cup R$ into the left set $L$ and the right set $R$ such that every edge of $G$ links a vertex of $L$ to a vertex of $R$. The strong **NP**-completeness of EWIS in bipartite graphs is straightforward since the *balanced biclique* problem (also called balanced complete bipartite subgraph) is **NP**-complete [20, 16]. This problem consists in deciding, given a bipartite graph $G = (L, R; E)$ and an integer $k$, if there exist $L' \subseteq L$ and $R' \subseteq R$ with $|L'| = |R'| = k$ such that the subgraph induced by $L' \cup R'$ is a complete bipartite subgraph (also called *biclique* of size $k$). In [16], a variation of this latter problem is introduced where we must have $|L'| = a$ and $|R'| = b$ (called the *biclique problem*). From an instance $(G, k)$ of balanced biclique, we introduce weight 1 on each vertex of $L$, weight $B = \max\{|L|, |R|\} + 1$ on each vertex of $R$, and we set $M = k + Bk$. It is clear that there exist an independent set in $(L, R; (L \times R) \setminus E)$ of weight $M$ if and only if there exists a balanced biclique in $(L, R; E)$ of size $k$.

This observation is strengthened in two ways in the following theorem.

**Theorem 4.** EWIS$_\alpha$ *is strongly* ***NP****-complete in bipartite graphs with maximum degree 3.*

*Proof.* The reduction is done from the decision clique set problem in regular graph which is known to be **NP**-complete, [20]. A clique $V^*$ is a subset of vertices of $G$ such that the subgraph induced by $V^*$ is complete. Let $G = (V, E)$ be a $\Delta$-regular graph of $n$ vertices and let $k$ be an integer. Wlog., assume that $\Delta < n - 1$ and $k > 0$ (thus, we can assume that $k < n$). We build the instance $I = (G', w)$ of EWIS where $G' = (L, R; E')$ is a bipartite graph as follows:

- For each vertex $v \in V$, we construct a gadget $H(v)$ which is a cycle of length $2\Delta$. Thus, it is a bipartite graph where the left set is $L_v = \{l_{1,v}, \ldots, l_{\Delta,v}\}$ and the right set is $R_v = \{r_{1,v}, \ldots, r_{\Delta,v}\}$. The weights are $w(l_{i,v}) = 1$ and $w(r_{i,v}) = n\Delta(\frac{2+n\Delta}{2})$ for $i = 1, \ldots, \Delta$. The gadget $H(v)$ is illustrated with Figure 1.

- For each edge $e \in E$, we construct a gadget $H(e)$ which is also a cycle of length $2n\Delta$. Thus, it is a bipartite graph where the left set is $L_e = \{l_{1,e}, \ldots, l_{n\Delta,e}\}$ and the right set is $R_e = \{r_{1,e}, \ldots, r_{n\Delta,e}\}$. The weights are $w(l_{i,e}) = 1$ and $w(r_{i,e}) = n\Delta(\frac{2+n\Delta}{2})$ for $i = 1, \ldots, n\Delta$. The gadget $H(e)$ is illustrated with Figure 2.

- We interconnect these gadgets by applying iteratively the following procedure. For each edge $e = [u, v] \in E$, we add one edge $[r_{i,u}, l_{1,e}]$ between gadgets $H(u)$, $H(e)$ and one edge $[r_{j,v}, l_{\frac{n\Delta}{2}+1,e}]$ between gadgets $H(v)$, $H(e)$ such that the vertices $r_{i,u}$, $r_{j,v}$ have degree 3.

It is clear that $G'$ is bipartite and the weights are polynomially bounded. Moreover, since $G$ is a $\Delta$-regular graph, we deduce that only the vertices of sets $R_v$, for $v \in V$ and $l_{1,e}$, $l_{\frac{n\Delta}{2}+1,e}$ for $e \in E$ have degree 3 in $G'$. All the others have degree 2.

We claim that there exist a clique $V^*$ of $G$ with a size at least $k$ if and only if there exist an independent set $S$ of $G'$ with weight exactly

$$B = \Delta k + n\Delta\frac{k(k-1)}{2} + n\Delta(\frac{2+n\Delta}{2})\left((n-k)\Delta + (\frac{n\Delta}{2} - \frac{k(k-1)}{2})n\Delta\right).$$

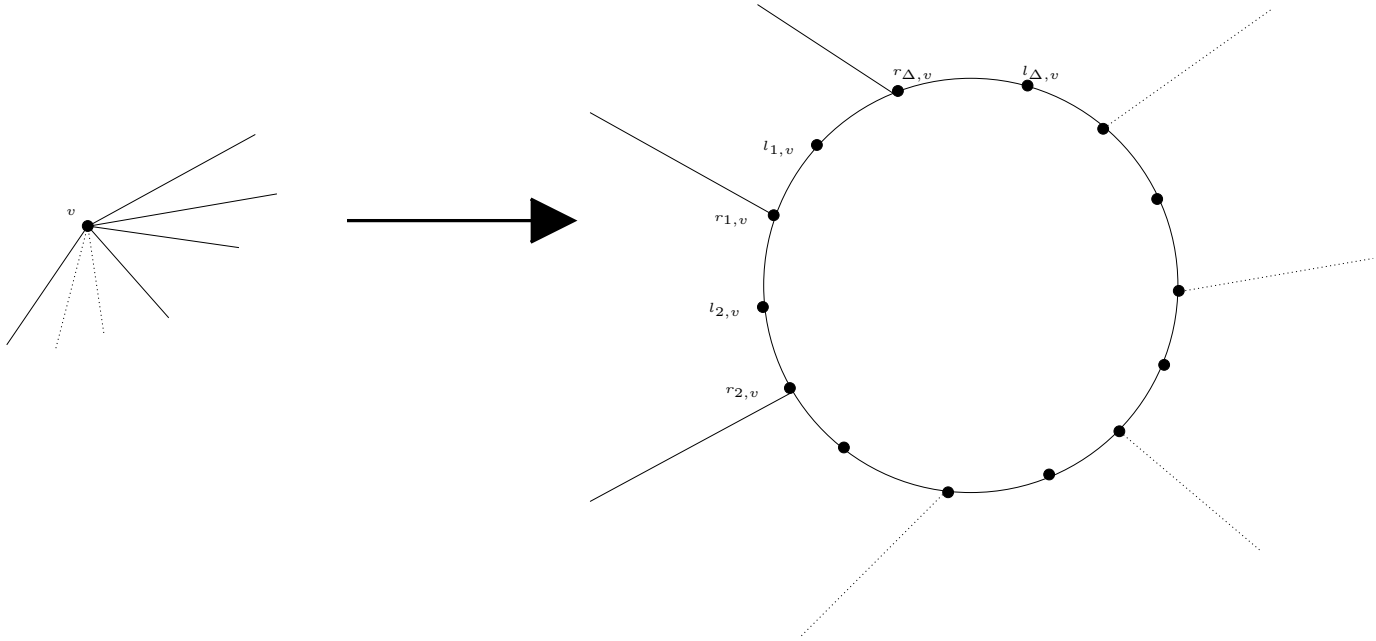


Figure 1: The gadget $H(v)$.

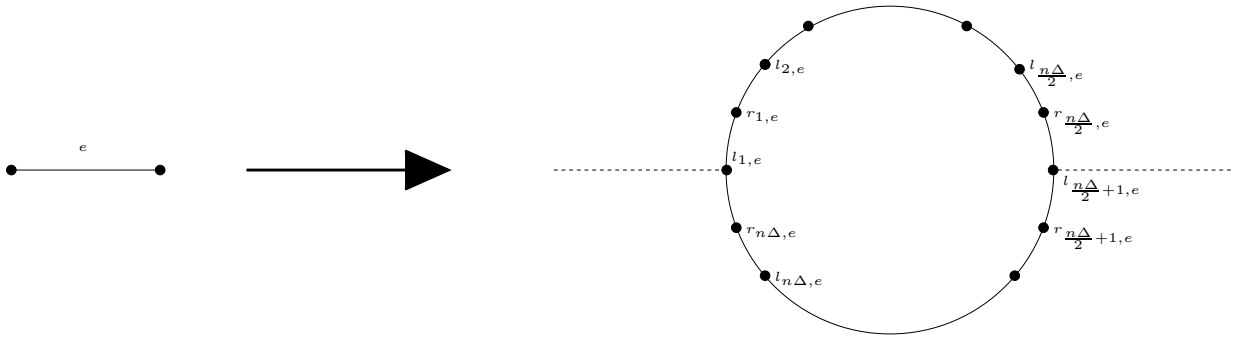


Figure 2: The gadget $H(e)$.

Let $V'$ be a clique of $G$ with $\mid V' \mid \geq k$ and consider a subclique $V^* \subseteq V'$ of size exactly $k$. We set $S = S_L \cup S_R$ with $S_L = \cup_{v \in V^*} L_v \cup_{e \in E(V^*)} L_e$ and $S_R = \cup_{v \in V \setminus V^*} R_v \cup_{e \in E \setminus E(V^*)} R_e$. One can easily verifies that $w(S) = B$ and that $S$ is an independent set of $G'$. Indeed, let us assume the reverse; thus, there exist $r_{i,v} \in S$ (and thus $R_v \subseteq S$), $l_{j,e} \in S$ (with $j = 1$ or $j = \frac{n\Delta}{2} + 1$) and $[r_{i,v}, l_{j,e}] \in E'$. By construction of $S$, we deduce that $e = [u, v] \in E(V^*)$ and then $L_v \subseteq S$, contradiction.

Conversely, let $S$ be an independent set of $G'$ with $w(S) = B$. We prove that $S$ is a maximum independent set of $G'$. Firstly, $|S| \leq |L| = |R| = n\Delta(\frac{n\Delta+2}{2})$ since $G'$ has a perfect matching (each gadget $H(v)$ and $H(e)$ contains a perfect matching). Secondly, since $w(v) = 1$ for $v \in L$ and $w(v) = |L| = |R| = n\Delta(\frac{n\Delta+2}{2})$ for $v \in R$, there is no possible compensation between $S \cap L$ and $S \cap R$. So, we obtain:

$$|S \cap L| = \Delta k + n\Delta \frac{k(k-1)}{2} \tag{1}$$

and

$$|S \cap R| = (n-k)\Delta + (\frac{n\Delta}{2} - \frac{k(k-1)}{2})n\Delta \quad (2)$$

Add inequality (1) to (2), the result follows. This implies in particular that for any vertex $v \in V$, either $L_v$ or $R_v$ is a subset of $S$. Moreover, the same property holds for any $e \in E$ (i.e., either $L_e$ or $R_e$ is a subset of $S$).

Now, we prove that there are exactly $k(\frac{k-1}{2})$ gadgets $H(e)$ with $L_e \subseteq S$. Assume that they are strictly less than it; then, $|S \cap L| \leq \Delta n + n\Delta(\frac{k(k-1)}{2} - 1)$. Combining this inequality with equality (1), we deduce that $k \leq 0$, contradiction. Now, assume that they are strictly more than it; using the same arguments that previously on $S \cap R$, we deduce this time that $|S \cap R| \leq (\frac{n\Delta}{2} - \frac{k(k-1)}{2})n\Delta$. Combining this with equality (2), we obtain $k \geq n$, which is impossible.

Thus, by construction: for any edge $e = [u, v] \in E$ with $L_e \subseteq S$, we must have $L_u \subseteq S$ and $L_v \subseteq S$. So, if we set $V^* = \{v \in V : L_v \subseteq S\}$, we deduce from previously $|V^*| \geq k$. Let us prove that $|V^*| = k$ and we will have necessarily $V^*$ is a clique of $G$. Assume that $|V^*| \geq k+1$; we obtain that $|S \cap R| \leq (n-k-1)\Delta + (\frac{n\Delta}{2} - \frac{k(k-1)}{2})n\Delta$. But, using equality (2), we obtain another contradiction (i.e., $\Delta \leq 0$). The proof is complete. □

As a corollary of Theorem 4, we can derive that the biclique problem remains **NP**-complete when the minimum degree of $G = (L, R; E)$ is $n-3$ where $|L| = |R| = n$. In this case, we replace any gadget $H(e)$ of Theorem 4 by a cycle of length $2n\Delta$ and we delete edges $[l_{i,u}, r_{1,e}]$ and $[l_{j,v}, r_{2,e}]$.

We also remark that Theorem 4 implies the strong **NP**-completeness of $\text{EWIS}_\alpha$ for perfect graphs, a well-known class of graphs where WIS is solvable in polynomial time.

### 3.2 A More General Hardness Result

We shall now strengthen the result of Theorem 3.1 to a more general setting, for hereditary subclasses of bipartite graphs in which no vertex degree exceeds 3. (A class of graphs is *hereditary* if it is closed under vertex deletion.) To this end, we first introduce some notations. We will denote the class of graphs containing no induced subgraphs from a set $\mathcal{F}$ by *Free*($\mathcal{F}$). Any graph in *Free*($\mathcal{F}$) will be called $\mathcal{F}$-free. Our hardness results will be expressed in terms of a parameter related to the set of forbidden induced subgraphs $\mathcal{F}$.

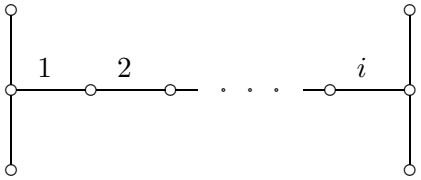


Figure 3: The graph $H_i$

For $i \geq 1$, let $H_i$ denote the graph depicted on Figure 3 and for $i \geq 3$, let $C_i$ denote the chordless cycle of length $i$. We associate to every graph $G$ a parameter $\kappa(G)$. If $G$ has a vertex of degree 4 or more, we define $\kappa(G)$ to be 0. Otherwise, let $\kappa(G)$ denote the minimum value of $k \geq 3$ such that $G$ contains an induced copy of either $C_k$ or $H_k$ (or $\infty$ if no such $k$ exists). Also, for a (possibly infinite) set of graphs $\mathcal{F}$, we define $\kappa(\mathcal{F}) = \sup\{\kappa(G) \ : \ G \in \mathcal{F}\}$.

With these definitions in mind, we can derive the following hardness result.

**Theorem 5.** *Let $\mathcal{F}$ be a set of graphs and let $\mathcal{G}$ be the set of $\mathcal{F}$-free bipartite graphs of maximum degree at most 3. If $\kappa(\mathcal{F}) < \infty$, then* $\text{EWIS}_\alpha$ *is strongly* ***NP****-complete in the class $\mathcal{G}$.*

*Proof.* The problem is clearly in **NP**. We show completeness in two steps. First, for $k \geq 3$, let $\mathcal{S}_k$ be the class of all bipartite $(C_3, \ldots, C_k, H_1, \ldots, H_k)$-free graphs of vertex degree at most 3, and let us show that for any fixed $k$, the problem is strongly **NP**-complete for graphs in $\mathcal{S}_k$. Let $(G, w, M)$ be an instance of $\text{EWIS}_\alpha$ where $G$ is a bipartite graph of maximum degree at most 3.

We can transform the graph $G$ in polynomial time to a vertex-weighted graph $G'$, as follows. Let $k' = \lceil \frac{k}{2} \rceil$. We replace each edge $e$ of $G$ by a path $P(e)$ on $2k' + 2$ vertices. Let $N = w(V) + 1$. We set the weights $w'$ of the endpoints of $P(e)$ equal to the weights of the corresponding endpoints of $e$, while each internal vertex of $P(e)$ gets weight $N$. It is easy to verify that $G'$ belongs to $\mathcal{S}_k$.

We claim that the answer to $\text{EWIS}_\alpha(G, w, M)$ is *yes* if and only if the answer to $\text{EWIS}_\alpha(G', w', M + mk'N)$ is *yes*, where $m = |E(G)|$.

One direction is immediate, as each maximum independent set of $G$ can be extended to a maximum independent set of $G'$, by simply adding $k'$ internal vertices of each newly added path. Doing so, the weight increases by $mk'N$.

Suppose now that the answer to $\text{EWIS}_\alpha(G', w', M + mk'N)$ is *yes*. Let $I'$ be a maximum independent set of $G'$ of weight $M + mk'N$. Since $I'$ is independent, it can contain at most $k'$ internal vertices of each newly added path. Therefore, for each $e \in E(G)$, the set $I'$ must contain *exactly* $k'$ internal vertices of $P(e)$ – otherwise its weight would be at most $W + (mk' - 1)N$, contradicting our choice of $N$.

Let $I$ denote the set, obtained from $I'$ by deleting the internal vertices of newly added paths. Then, $I$ is an independent set of $G$. Indeed, if $e = [u, v] \in E(G)$ for some $u, v \in I$, then $I'$ can contain at most $k' - 1$ internal vertices of $P(e)$, contradicting the above observation. Also, it is easy to see that $I$ is a maximum independent set of $G$. Finally, as the weight of $I$ is exactly $M$, we conclude that the answer to $\text{EWIS}_\alpha(G, w, M)$ is *yes*.

This shows that $\text{EWIS}_\alpha$ is strongly **NP**-complete in the class $\mathcal{S}_k$. To prove strong **NP**-completeness of the problem in the class $\mathcal{G}$, we now show that the class $\mathcal{G}$ contains all graphs in $\mathcal{S}_k$, for $k := \max\{3, \kappa(\mathcal{F})\}$. Let $G$ be a graph from $\mathcal{S}_k$. Assume that $G$ does not belong to $\mathcal{G}$. Then $G$ contains a graph $A \in \mathcal{F}$ as an induced subgraph. From the choice of $G$ we know that $A$ belongs to $\mathcal{S}_k$, but then $k < \kappa(A) \leq \kappa(\mathcal{F}) \leq k$, a contradiction. Therefore, $G \in \mathcal{G}$ and the theorem is proved. □

# 4 Polynomial Results

In this section, we identify several graph classes where EWIS and $\text{EWIS}_\alpha$ can be solved in pseudo-polynomial time. By part $(ii)$ of Lemma 1, it suffices to develop pseudo-polynomial solutions for EWIS. For this purpose, the following version of the problem turns out to be most appropriate: Given a vertex-weighted graph $(G, w)$ and a target $M$, compute the solutions to all $M$ instances of EWIS with varying target $k \in [M]$. That is, the solution to the problem is given by the vector $(\text{EWIS}(G, w, k) : k \in [M]) \in \{yes, no\}^{[M]}$.

First, we show that without loss of generality, we may restrict our attention to connected graphs. To do so, we consider the following generalization of the subset sum problem, a typical example of an **NP**-complete problem that can be solved in pseudo-polynomial time by dynamic programming.

**GENERALIZED SUBSET SUM (GSS)**
**Instance**: Nonempty sets of positive integers $A_1, \ldots, A_n$ and a positive integer $b$.

**Question**: Is there a nonempty subset $S$ of $[n]$ and a mapping $a : S \to \cup_{i \in S} A_i$ such that $a(i) \in A_i$ for all $i \in S$, and $\sum_{i \in S} a(i) = b$?

It is straightforward to extend the dynamic programming solution for subset sum to one for generalized subset sum (a proof is given in the appendix).

**Lemma 6.** *There is an $O(nb^2)$ dynamic programming algorithm that computes the set $B$ of all values $b' \in [b]$ such that there is a set $S$ and a mapping $a$ (as above) such that $\sum_{i \in S} a(i) = b'$.*

It now follows immediately that in order to solve EWIS, it suffices to solve the problem for connected graphs.

**Corollary 7.** *Let $(G, w, M)$ be an instance of* EWIS*, and let $C_1, \ldots, C_r$ be the connected components of $G$. Suppose that for each $i \in [r]$, the set of solutions* $(\text{EWIS}(C_i, w, k) : k \in [M])$ *for $C_i$ is given. Then, we can compute the set of solutions* $(\text{EWIS}(G, w, k) : k \in [M])$ *for $G$ in time $O(rM^2)$.*

*Proof.* It suffices to observe that for every $k \in [M]$, the solution to $\text{EWIS}(G, w, k)$ is *yes* if and only if the solution to the GSS problem on the instance $(A_1, \ldots, A_r; k)$ is *yes*, where $A_i$ denotes the set of all values $k' \in [M]$ such that the solution to $\text{EWIS}(C_i, w, k')$ is *yes*. □

In view of the connection between the weighted independent set problem and its exact counterpart (cf. part $(i)$ of Lemma 1), the following question arises naturally: Which of the polynomial-time solutions for the WIS problem on restricted input graphs can be extended to solutions for the EWIS problem? Most commonly used approaches to the WIS problem in particular graph classes can be summarized as follows:

- **Dynamic programming.** Typically, solutions based on dynamic programming rely on structural properties of graphs in a given class. Example include interval graphs [34], AT-free graphs [12], distance-hereditary graphs [4, 14], circle graphs [36], graphs of tree-width at most $k$ [2], graphs of clique-width at most $k$ [13], etc. (We do not define these graph classes here; the interested reader is referred to the comprehensive survey [11].)
- **Decomposition by clique separators [37, 39].** This approach can be used for example to develop a simple solution to the WIS problem in chordal graphs.
- **Enumeration of all maximal independent sets [38].** This approach becomes feasible whenever the graphs in a given class contain only polynomially many maximal independent sets. This is the case for instance for $mK_2$-free graphs (for every $m \geq 1$) [1, 3, 33].
- **Modular decomposition.** Besides other results [9], this approach has been used to solve the WIS problem in certain subclasses of $P_5$-free and fork-free graphs [10].

It turns out that almost all of these solutions to the WIS problem can be extended to solutions for the EWIS problem, the only exception being the decomposition by clique separators. Nevertheless, for chordal graphs a different solution for the EWIS problem can be developed based on their clique tree representation and using a set of identities developed by Okamoto *et al.* in [31].

In the following theorem, we summarize our results about pseudo-polynomial solutions for the problem of finding independent sets of given weight. Most proofs and algorithms can be found in the appendix.

**Theorem 8.** *The problems* EWIS *and* $\text{EWIS}_\alpha$ *can be solved in pseudo-polynomial time in each of the following graph classes: interval graphs, chordal graphs, AT-free graphs, $(claw, net)$-free graphs, distance-hereditary graphs, circle graphs, $mK_2$-free graphs, graphs of tree-width at most $k$, and graphs of clique-width at most $k$.*

**Remark.** For graphs of clique-width at most $k$, an alternative to a direct solution to the EWIS problem can be obtained by adapting the general result of Courcelle *et al.* [13] which shows that every optimization problem expressible in monadic second order logic with quantification over the vertices and vertex sets can be solved in linear time on graphs of bounded clique-width. Indeed, it is not hard to adapt the notions from [13] to the "exact" setting and to modify the proofs therein to show that, roughly speaking, *every "exact" problem expressible in monadic second order logic with quantification over the vertices and vertex sets can be solved in pseudo-polynomial time for graphs of bounded clique-width.*

In the remainder of this section, we recall the notion of the modular decomposition of a graph and show how it can be applied to the EWIS problem. The idea of modular decomposition has been first described in the 1960s by Gallai [19], and also appeared in the literature under various other names such as *prime tree decomposition* [18], *X-join decomposition* [22], or *substitution decomposition* [28]. This technique allows one to reduce many graph problems from arbitrary graphs to the so-called *prime* graphs. For an overview of the applications of modular decomposition to combinatorial optimization and other problems, together with a general algebraic decomposition theory, we refer the reader to the paper of Möhring and Rademacher [29].

Let $G = (V, E)$ be a graph, $U$ a subset of $V$ and $x$ a vertex of $G$ outside $U$. We say that $x$ *distinguishes* $U$ if $x$ has both a neighbor and a non-neighbor in $U$. A subset $U \subset V(G)$ is called a *module* in $G$ if it is indistinguishable for the vertices outside $U$. A module $U$ is *nontrivial* if $1 < |U| < |V|$, otherwise it is *trivial*. A *prime* graph is one with only trivial modules.

Modular decomposition provides a reduction of many graph problems from a graph $G$ to the graph $G^0$ obtained from $G$ by contracting each maximal module to a single vertex. We formally describe this reduction for the EWIS problem in the recursive procedure MODULAR_EWIS$(G, W, M)$ below. It turns out that in order to apply this decomposition to the EWIS problem, we need to relax the problem so that each vertex of the input graph is equipped with a nonempty set of possible weights (instead of just a single one). We name this generalized problem GEWIS. When all sets are singletons, the problem coincides with the original EWIS problem.

**GENERALIZED EXACT WEIGHTED INDEPENDENT SET (GEWIS)**
**Instance**: An ordered triple $(G, W, M)$, where $G = (V, E)$ is a graph, $M$ is a positive integer and $W = (W_v : v \in V)$ with $W_v \subseteq [M]$ for all $v \in V$ is the collection of possible weights for each vertex of $G$.
**Question**: Is there an independent set $I$ of $G$ and a mapping $w : I \to [M]$ such that $w(v) \in W_v$ for all $v \in I$, and $\sum_{v \in I} w(v) = M$?

*Algorithm* MODULAR_EWIS$(G, W, M)$

**Input:** An ordered triple $(G, W, M)$, where $G = (V, E)$ is a graph, $M$ is a positive integer and $W = (W_v : v \in V)$ with $W_v \subseteq [M]$ for all $v \in V$ is the collection of possible weights for each vertex of $G$.

**Output:** $(\mathrm{EWIS}(G, W, k) : k \in [M])$

1. If $|V| = 1$, say $V = \{v\}$, set, for each $k \in [M]$, $\mathrm{EWIS}(G, W, k) = \begin{cases} yes, & \text{if } k \in W_v; \\ no, & \text{otherwise} \end{cases}$ and go to step 7.
2. If $G$ is disconnected, partition it into components $\mathcal{M}_1, \ldots, \mathcal{M}_r$, and go to step 5.
3. If co-$G$ is disconnected, partition $G$ into co-components $\mathcal{M}_1, \ldots, \mathcal{M}_r$, and go to step 5.
4. If $G$ and co-$G$ are connected, partition $G$ into maximal modules $\mathcal{M}_1, \ldots, \mathcal{M}_r$.

5. For all $j \in [r]$, let $(EWIS(G[\mathcal{M}_j], W, k) : k \in [M]) =$ MODULAR_EWIS$(G[\mathcal{M}_j], W, M)$. Construct a graph $G^0$ from $G$ by contracting each $\mathcal{M}_j$ (for $j \in [r]$) to a single vertex, and assign to that vertex the set of weights $W_{\mathcal{M}_j} = \{k \in [M] : \text{EWIS}(G[\mathcal{M}_j], W, k) = yes\}$.

6. For each $k \in [M]$, let $\text{EWIS}(G, W, k) = \text{EWIS}(G^0, (W_{\mathcal{M}_j} : j \in [r]), k)$.

7. Return $(\text{EWIS}(G, W, k) : k \in [M])$ and stop.

For each input graph, at most one of the steps 2-4 is performed. (At most one among $\{G$, co-$G\}$ is disconnected; moreover, if $G$ and co-$G$ are both connected, then the maximal modules of $G$ are pairwise disjoint.) Observe that the graph $G^0$ constructed in step 5 of the algorithm is either an edgeless graph, a complete graph, or a prime graph. Therefore, the modular decomposition approach reduces the problem from a graph to its prime induced subgraphs.

The correctness of the procedure is straightforward: every independent set $I$ of $G$ consists of pairwise disjoint independent sets in the subgraphs of $G$ induced by $\mathcal{M}_1, \ldots, \mathcal{M}_r$; moreover, those $\mathcal{M}_i$'s that contain a vertex from $I$ form an independent set in $G^0$. And conversely, for every independent set $I^0$ in $G^0$ and every choice of independent sets $\{I_j : j \in I^0\}$ with $I_j$ independent in $G[\mathcal{M}_j]$, the set $\cup_{j \in [r]} I_j$ is independent in $G$. The following theorem answers the question on the complexity of such a reduction.

**Theorem 9.** *Let $\mathcal{G}$ be a class of graphs and $\mathcal{G}^*$ the class of all prime induced subgraphs of the graphs in $\mathcal{G}$. If there is a $p \geq 1$ and a $q \geq 2$ such that* GEWIS *can be solved for graphs in $\mathcal{G}^*$ in time $O(M^q n^p)$, then* GEWIS *can be solved for graphs in $\mathcal{G}$ in time $O(M^q n^p + m)$.*

We omit the proof as it is a straightforward adaptation of the proof of the analogous result for the WIS problem (see e.g. [27]).

Theorem 9 leads to pseudo-polynomial-time solutions to EWIS in several subclasses of $P_5$-free and *fork*-free graphs. The results are summarized in the following theorem; all graphs mentioned in the theorem or its proof (see appendix) are depicted in Figure 4.

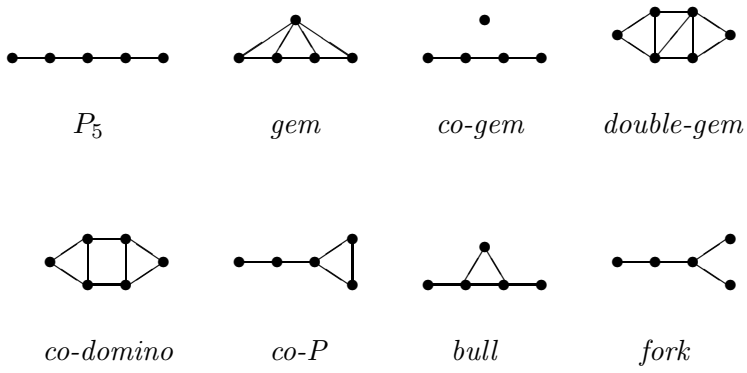


Figure 4: Some 5- and 6-vertex graphs

**Theorem 10.** EWIS *is solvable in pseudo-polynomial time in each of the following classes: ($P_5$, double-gem, co-domino)-free graphs (and their subclass ($P_5$, co-P)-free graphs), (bull, fork)-free graphs, (co-P, fork)-free graphs, and ($P_5$, fork)-free graphs.*

**Remark.** Due to the relation between the exact perfect matching problem and the exact weighted maximum independent set problem, every polynomial result for EWIS and EWIS$_\alpha$ yields a polynomial result for the exact perfect matching problem. Whenever EWIS or EWIS$_\alpha$

are (pseudo-)polynomially solvable for a class of graphs $\mathcal{G}$, the exact perfect matching problem is (pseudo-)polynomially solvable for graphs in the set $\{G : L(G) \in \mathcal{G}\}$. For example, since a set $\mathcal{G}$ of graphs has bounded tree-width if and only if $L(\mathcal{G})$ has bounded clique-width [21], it follows that the exact perfect matching problem is solvable in pseudo-polynomial time for graphs of bounded tree-width.

# 5 Concluding Remarks

In view of the unknown complexity of the exact perfect matching problem, the problem of determining the complexity status of EWIS and EWIS$_\alpha$ is of particular interest for line graphs of bipartite graphs, and their subclasses and superclasses. Line graphs of bipartite graphs form a hereditary class of graphs, and they can be characterized in terms of forbidden induced subgraphs as follows: A graph $G$ is the line graph of a bipartite graph if and only $G$ is $\mathcal{F}$-free, where $\mathcal{F} = \{\mathit{claw}, \mathit{diamond}, C_5, C_7, \ldots\}$, where a *diamond* is the graph obtained by deleting a single edge from a complete graph on 4 vertices [35]. Using this characterization and some the results obtained above, we now show that the class $L(\mathit{Bip})$ of line graphs of bipartite graphs is sandwiched between two graph classes for which the complexity of EWIS$_\alpha$ is known, and whose (infinite) sets of forbidden induced subgraphs differ only in two graphs.

Replacing the diamond in the above characterization of line graphs of bipartite graphs by its subgraph $C_3$ results in a smaller class of $(\mathit{claw}, C_3, C_5, C_7, \ldots)$-free graphs. This is precisely the class of bipartite graphs of maximum degree at most 2, which means that every connected component of such a graph is either path or an even cycle. The tree-width of such graphs is at most 2, hence the problem is solvable in pseudo-polynomial time in this class. On the other hand, if we replace the $\mathit{claw} = K_{1,3}$ with $K_{1,4}$ in the above characterization of $L(\mathit{Bip})$, we obtain a class of graphs that strictly contains line graphs of bipartite graphs. This class of $(K_{1,4}, \mathit{diamond}, C_5, C_7, \ldots)$-free graphs also contains the class of $(K_{1,4}, C_3, C_5, C_7, \ldots)$-free graphs, which is precisely the class of bipartite graphs of maximum degree at most 3. By Theorem 5 from Section 3.1, the problem EWIS$_\alpha$ is hard for this class, and hence also for the larger class of $(K_{1,4}, \mathit{diamond}, C_5, C_7, \ldots)$-free graphs.

To summarize, the class $L(\mathit{Bip})$ of line graphs of bipartite graphs is sandwiched between two graph classes for which the complexity of EWIS$_\alpha$ is known, as the following diagram shows:

$$\begin{array}{ccccc} \mathit{Free}(\{\mathit{claw}, C_3, C_5, C_7, \ldots\}) & \subset & L(\mathit{Bip}) & \subset & \mathit{Free}(\{K_{1,4}, \mathit{diamond}, C_5, C_7, \ldots\}) \\ \text{pseudo-polynomial} & & ??? & & \text{strongly NP-hard} \end{array}$$

It would be interesting to identify further sub- and superclasses of line graphs of bipartite graphs where EWIS$_\alpha$ is solvable in pseudo-polynomial time or strongly **NP**−hard, respectively. Another related question would be to determine the complexity status of EWIS and EWIS$_\alpha$ in certain subclasses of perfect and bipartite graphs such as weakly chordal graphs and their subclass chordal bipartite graphs.

# References


[1] V.E. Alekseev, "On the number of maximal independent sets in graphs from hereditary classes," Combinatorial-algebraic methods in discrete optimization, University of Nizhny Novgorod, 1991, pp. 58 (in Russian).

[2] S. Arnborg and A. Proskurowski, "Linear time algorithms for NP-hard problems restricted to partial $k$-trees," *Discrete Appl. Math.* **23** (1989), 11–24.

[3] E. Balas and C.S. Yu, "On graphs with polynomially solvable maximum-weight clique problem," *Networks* **19** (1989) 247-253.

[4] H.-J. Bandelt and H.M. Mulder, "Distance-hereditary graphs," *J. Combin. Theory Ser. B* **41** (1986) 182–208.

[5] F. Barahona and W.R. Pulleyblank, "Exact arborescences, matchings, and cycles," *Discrete Appl. Math.* **16** (1987) 91-99.

[6] C. Berge, *Graphs and hypergraphs.* North Holland, Amsterdam, 1973.

[7] J. Błażewicz, P. Formanowicz, M. Kasprzak, P. Schuurman and G. Woeginger, "A polynomial time equivalence between DNA sequencing and the exact perfect matching problem," *Discrete Optim.* (2007) 154–162.

[8] A. Brandstädt and F.F. Dragan, "On linear and circular structure of (claw, net)-free graphs," *Discrete Appl. Math.* **129** (2003) 285–303.

[9] A. Brandstädt, C.T. Hoàng and J.-M. Vanherpe, "On minimal prime extensions of a four-vertex graph in a prime graph," *Discrete Math.* **288** (2004) 9–17.

[10] A. Brandstädt, V.B. Le and H.N. de Ridder, "Efficient robust algorithms for the maximum weight stable set problem in chair-free graph classes," *Inform. Process. Lett.* **89** (2004) 165-173.

[11] A. Brandstädt, V.B. Le and J. Spinrad, *Graph classes: a survey.* SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia, PA, 1999.

[12] H. Broersma, T. Kloks, D. Kratsch and H. Müller, "Independent sets in asteroidal triple-free graphs," *SIAM J. Discrete Math.* **12** (1999) 276–287.

[13] B. Courcelle, J.A. Makowsky and U. Rotics, "Linear Time Solvable Optimization Problems on Graphs of Bounded Clique-Width," *Theory Comput. Systems* **33** (2000) 125–150.

[14] O. Cogis and E. Thierry, "Computing maximum stable sets for distance-hereditary graphs," *Discrete Optim.* **2** (2005) 185–188.

[15] G. Damiand, M. Habib and C. Paul, "A simple paradigm for graph recognition: application to cographs and distance hereditary graphs." Combinatorics and computer science (Palaiseau, 1997). *Theoret. Comput. Sci.* **263** (2001) 99–111.

[16] M. Dawande, P. Keskinocak, J.M. Swaminathan and S. Tayur, "On Bipartite and Multipartite Clique Problems," *J. Algorithms* **41** (2001) 388–403.

[17] V.G. Deĭneko and G.J. Woeginger, "On the robust assignment problem under a fixed number of cost scenarios," *Oper. Res. Lett.* **34** (2006) 175–179.

[18] A. Ehrenfeucht and G. Rozenberg, "Primitivity is hereditary for 2-structures," *Theoret. Comput. Sci.* 70 (1990) 343–358.

[19] T. Gallai, "Transitiv orientierbare graphen," *Acta Math. Acad. Sci. Hungar.* **18** (1967) 25–66.

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

[21] F. Gurski and E. Wanke, "Line graphs of bounded clique-width," *Discrete Appl. Math.* to appear, 2007.

[22] M. Habib and M.C. Maurer, "On the X-join decomposition for undirected graphs," *Discrete Appl. Math.* 1 (1979) 201–207.

[23] C.T. Hoáng and B. Reed, "Some classes of perfectly orderable graphs," *J. Graph Theory* **13** (1989) 445-463.

[24] W.-L. Hsu and T.-H. Ma, "Fast and simple algorithms for recognizing chordal comparability graphs and interval graphs," *SIAM J. Comput.* **28** (1999), 1004-1020.

[25] A.V. Karzanov, "Maximum matching of given weight in complete and complete bipartite graphs," *Cybernetics* **23** (1987) 8-13; translation from *Kibernetika* **1** (1987) 7-11.

[26] M. Leclerc, "Polynomial time algorithms for exact matching problems," Masters Thesis, University of Waterloo, Waterloo, 1986.

[27] V.V. Lozin and M. Milanič, "A polynomial algorithm to find an independent set of maximum weight in a fork-free graph," *J. Discrete Algorithms* (2008), to appear. doi:10.1016/j.jda.2008.04.001.

[28] R.H. Möhring, "Algorithmic aspects of comparability graphs and interval graphs," in: I. Rival (Ed.), *Graphs and Orders*, D. Reidel, Boston, 1985, 41–101.

[29] R.H. Möhring and F.J. Radermacher, Substitution decomposition for discrete structures and connections with combinatorial optimization. Algebraic and combinatorial methods in operations research, 257–355, North-Holland Math. Stud., 95, North-Holland, Amterdam, 1984.

[30] K. Mulmuley, U. Vazirani and V.V. Vazirani, "Matching is as easy as matrix inversion," *Combinatorica* **7** (1987) 105-113.

[31] Y. Okamoto, T. Uno and R. Uehara, "Linear-time counting algorithms for independent sets in chordal graphs," Graph-theoretic concepts in computer science, 433–444, *Lecture Notes in Comput. Sci.* **3787**, Springer, Berlin, 2005.

[32] C.H. Papadimitriou and M. Yannakakis, "The complexity of restricted spanning tree problems," *J. ACM* **29** (1982) 285-309.

[33] E. Prisner, "Graphs with few cliques," Graph Theory, Combinatorics, and Algorithms, Vol. 1, 2 (Kalamazoo, MI, 1992), 945-956, Wiley-Interscience Publ., Wiley, New York, 1995.

[34] J. Ramalingam and C. Pandu Rangan, "A unified approach to domination problems on interval graphs," *Inform. Process. Lett.* **27** (1988) 271-274.

[35] W. Staton and G.C. Wingard, "On line graphs of bipartite graphs," *Util. Math.* **53** (1998) 183–187.

[36] K.J. Supowit, "Finding a maximum planar subset of a set of nets in a channel," *IEEE Trans. on CAD of ICAS* **CAD-6** (1987) 93–94.

[37] R.E. Tarjan, "Decomposition by clique separators," *Discrete Math.* **55** (1985) 221–232.

[38] S. Tsukiyama, M. Ide, H. Ariyoshi and I. Shirakawa, "A new algorithm for generating all the maximal independent sets," *SIAM J. Comput.* **6** (1977) 505-517.

[39] S.H. Whitesides, An algorithm for finding clique cut-sets, *Inform. Process. Lett.* 12 (1981) 31–32.

# Appendix

We present some of the missing proofs and algorithms.

**Lemma 6.** *There is an $O(nb^2)$ dynamic programming algorithm that computes the set $B$ of all values $b' \in [b]$ such that there is a set $S$ and a mapping $a$ (as above) such that $\sum_{i \in S} a(i) = b'$.*

*Proof.* The proof is by induction on $n$. The statement clearly holds for $n = 1$, as in this case we have $B = \{b' \in A_1 : b' \leq b\}$.

Suppose now that $n > 1$. Let $I = (A_1, \ldots, A_n; b)$ be an instance of the GSS problem. Let $B'$ be the inductively constructed set of all possible values of $b' \in [b]$ such that the solution to the GSS problem on the instance $(A_1, \ldots, A_{n-1}; b')$ is *yes*. By induction, the set $B'$ was constructed in time $O((n-1)b^2)$.

Let $\beta \in [b]$. Then, $\beta$ will belong to $B$, i.e., the solution to the GSS, given $(A_1, \ldots, A_n; \beta)$, will be *yes*, if and only if either $\beta \in B'$, or we can write $\beta$ as $\beta = b' + a_n$ for some $b' \in B'$ and $a_n \in A_n$. In other words, $B = B' \cup B''$, where $B''$ denotes the set of all such sums: $B'' = \{b' + a_n : b' \in B', a_n \in A_n, b' + a_n \leq b\}$.

The set $B''$ can be constructed in time $O(b^2)$. Adding this time complexity to the time $O((n-1)b^2)$ needed to construct $B'$ proves the lemma. □

## Pseudo-polynomial-time solutions to EWIS

### Interval graphs

Interval graphs are intersection graphs of intervals on the real line, and many optimization problems can be solved by dynamic programming on these graphs. A representation of interval graphs that is particularly suitable for the EWIS problem is the following. It has been shown by Ramalingam and Pandu Rangan [34] that a graph $G = (V, E)$ is interval if and only if it admits a vertex ordering $(v_1, \ldots, v_n)$ such that for all triples $(r, s, t)$ with $1 \leq r < s < t \leq n$, the following implication is true:

$$\text{if } \ [v_r, v_t] \in E \ \text{ then } \ [v_s, v_t] \in E\,.$$

Moreover, such an ordering of an interval graph can be found in time $O(n + m)$. Based on this ordering, we can prove the following statement.

**Theorem 11.** EWIS *admits an $O(Mn + m)$ algorithm for interval graphs.*

*Proof.* Let $(v_1, \ldots, v_n)$ be a vertex ordering such that $[v_s, v_t] \in E$, whenever $[v_r, v_t] \in E$, for all triples $(r, s, t)$ with $1 \leq r < s < t \leq n$.

For every $i \in [n]$, let $G_i$ denote the subgraph of $G$ induced by $\{v_1, \ldots, v_i\}$ (also, let $G_0$ be the empty graph). Then, for every $i \in [n]$, either there is a $j = j(i)$ such that $N_{G_i}(v_i) = \{j, j+1, \ldots, i-1\}$, or $N_{G_i}(v_i) = \emptyset$ (in which case let us define $j(i) = i$). Now, if $I$ is an independent set of $G_i$, then either $v_i \in I$ (in which case $I \backslash \{v_i\}$ is an independent set of $G_{j(i)-1}$), or $v_i \notin I$ (in which case $I$ is an independent set of $G_{i-1}$). This observation is the key to the following simple $O(Mn + m)$ dynamic programming solution to the EWIS problem on interval graphs.

*Step 1.* Find a vertex ordering $(v_1, \ldots, v_n)$ as above.
*Step 2.* Set EWIS$(G_0, w, k)$ to *no* for all $k \in [M]$.
*Step 3.* For $i = 1, \ldots, n$, do the following:

*3.1.* Find $j \in [i]$ such that $N_{G_i}(v_i) = \{j, j+1, \ldots, i-1\}$.

*3.2.* For $k \in [M]$, do the following:
If $k = w(v_i)$, set EWIS$(G_i, w, k)$ to *yes*.
If $k < w(v_i)$, set EWIS$(G_i, w, k)$ to EWIS$(G_{i-1}, w, k)$.
If $k > w(v_i)$, set EWIS$(G_i, w, k)$ to *yes* if at least one of the solutions to EWIS$(G_{j(i)-1}, w, k - w(v_i))$ and EWIS$(G_{i-1}, w, k)$ is *yes*, and to *no* otherwise.

*Step 4.* Output the solution to EWIS$(G_n, w, M)$. □

### AT-free and $(claw, net)$-free graphs

A triple $\{x, y, z\}$ of pairwise non-adjacent vertices in a graph $G$ is an *asteroidal triple* if for every two of these vertices there is a path between them avoiding the closed neighborhood of the third. Formally, $x$ and $y$ are in the same component of $G - N[z]$, $x$ and $z$ are in the same component of $G - N[y]$, and $y$ and $z$ are in the same component of $G - N[y]$.[1] A graph is called *AT-free* if it has no asteroidal triples.

Our dynamic programming algorithm that solves EWIS for AT-free graphs is based on the dynamic programming approach to the WIS problem in AT-free graphs, developed by Broersma *et al.* in [12]. Let us start with a definition.

**Definition 1.** *Let $x$ and $y$ be two distinct nonadjacent vertices of an AT-free graph $G$. The interval $I(x, y)$ is the set of all vertices $z$ of $V(G)\backslash\{x, y\}$ such that $x$ and $z$ are in one component of $G - N[y]$, and $z$ and $y$ are in one component of $G - N[x]$.*

Now, we recall some structural results from [12].

**Theorem 12** ([12])**.** *Let $I = I(x, y)$ be a nonempty interval of an AT-free graph $G$, and let $s \in I$. Then there exist components $C_1^s, \ldots, C_t^s$ of $G - N[s]$ such that the components of $I\backslash N[s]$ are precisely $I(x, s)$, $I(s, y)$, and $C_1^s, \ldots, C_t^s$.*

**Theorem 13** ([12])**.** *Let $G$ be an AT-free graph, let $C$ be a component of $G - N[x]$, let $y \in C$, and let $D$ be a component of the graph $C - N[y]$. Then $N[D] \cap (N[x]\backslash N[y]) = \emptyset$ if and only if $D$ is a component of $G - N[y]$.*

**Theorem 14** ([12])**.** *Let $G$ be an AT-free graph, let $C$ be a component of $G - N[x]$, let $y \in C$, and let $C'$ be the component of $G - N[y]$ that contains $x$. Let $B_1, \ldots, B_l$ denote the components of the graph $C - N[y]$ that are contained in $C'$. Then $I(x, y) = \cup_{i=1}^{l} B_i$.*

We will also need the following general statement.

**Observation 15.** *Let $(G, w)$ be a weighted graph. Then, the solution to* EWIS$(G, w, k)$ *is yes if and only if there is a vertex $x \in V(G)$ such that the solution to* EWIS$(G - N(x), w, k)$ *is yes.*

Combining this observation with Theorems 13 and 14, we obtain the following lemma.

**Lemma 16.** *Let $(G, w)$ be a weighted AT-free graph, $G = (V, E)$. Let $x \in V$ and let $C$ be a component of $G - N[x]$. For a vertex $y$ of $C$, let $C_y$ denote the subgraph of $G$ induced by $C - N(y)$. Then, the solution to* EWIS$(C, w, k)$ *is yes if and only if there is a vertex $y \in C$ such that the solution to* EWIS$(C_y, w, k)$ *is yes. Moreover, the connected components of such a $C_y$ are precisely $\{y\}$, $I(x, y)$, and the components of $G - N[y]$ contained in $C$.*

[1] Recall that the closed neighborhood of $x$ is defined as $N[x] = N(x) \cup \{x\}$.

Similarly, using Theorem 12 we obtain the following conclusion.

**Lemma 17.** *Let $(G, w)$ be a weighted AT-free graph, $G = (V, E)$. Let $I = I(x, y)$ be an interval of $G$. If $I = \emptyset$, then the solution to EWIS$(G[I], w, k)$ is yes if and only if $k = 0$. Otherwise, let us denote by $I_s$ the subgraph of $G$ induced by $I - N(s)$, for all $s \in I$. Then, the solution to* EWIS$(I, w, k)$ *is yes if and only if there is a vertex $s \in I$ such that the solution to* EWIS$(I_s, w, k)$ *is yes. Moreover, the connected components of such an $I_s$ are precisely $\{s\}$, $I(x, s)$, $I(s, y)$, and the components of $G - N[s]$ contained in $I$.*

**Theorem 18.** EWIS *admits a pseudo-polynomial algorithm for AT-free graphs.*

**Proof.** It follows from the above discussion that the following pseudo-polynomial algorithm correctly solves the problem.

*Step 1.* For every $x \in V$ compute all components of $G - N[x]$.
*Step 2.* For every pair of nonadjacent vertices $x, y \in V(G)$ compute the interval $I(x, y)$.
*Step 3.* Sort all the components and intervals according to nonincreasing number of vertices.
*Step 4.* In the order of Step 3, compute the solutions to EWIS$(C, w, k)$, for each component $C$ (for all $k \in \{0, 1, \ldots, w(C)\}$) and the solutions to EWIS$(I, w, k)$ for each interval $I$ (for all $k \in \{0, 1, \ldots, w(I)\}$). To compute the solutions to EWIS$(C, w, k)$ for a component $C$, first compute the solutions to EWIS$(C - N(y), w, k)$, for all $y \in C$, by applying Lemma 16 and Corollary 7. Similarly, to compute the solutions to EWIS$(I, w, k)$ for an interval $I$, first compute the solutions to EWIS$(I - N(s), w, k)$, for all $s \in I$, by applying Lemma 17 and Corollary 7.
*Step 5.* Compute the solution to EWIS$(G, w, k)$ using Observation 15 and Corollary 7. □

In [8], it is shown that for every vertex $v$ of a $(claw, net)$-free graph $G$, the non-neighborhood of $v$ in $G$ is AT-free. Thus, Theorem 18 immediately implies the following result.

**Corollary 19.** EWIS *admits a pseudo-polynomial algorithm for $(claw, net)$-free graphs.*

### Distance-hereditary graphs

A graph is *distance hereditary* if the distance between any two connected vertices (that is, vertices in the same connected component) is the same in every induced subgraph in which they remain connected.[2] Bandelt and Mulder provided in [4] a *pruning sequence* characterization of distance-hereditary graphs: whenever a graph contains a vertex of degree one, or a vertex with a twin (another vertex sharing the same neighbors), remove such a vertex. A graph is distance-hereditary if and only if it the application of such vertex removals results in a single-vertex graph.

A pruning sequence of a distance-hereditary graph can be computed in linear time [15] and can be useful for algorithmic developments on distance-hereditary graphs. A solution to the WIS problem in distance-hereditary graphs based on the pruning sequence characterization has been developed by Cogis and Thierry in [14]. It turns out that their approach can be generalized in order to solve the exact version of the problem.

**Theorem 20.** EWIS *admits an $O(M^2n + m)$ algorithm for distance-hereditary graphs.*

### Circle graphs

*Circle graphs* are the intersection graphs of chords on a circle. Our algorithm for EWIS on circle graphs is based on the dynamic programming solution for the IS problem, developed by Supowit in [36].

[2] The distance between two vertices $u$ and $v$ in a connected graph $G$ is the length (i.e., the number of edges) of a shortest path connecting them.

**Theorem 21.** EWIS *admits an* $O(M^2n^2)$ *algorithm for circle graphs.*

*Proof.* Consider a finite set of $N$ chords on a circle. We may assume without loss of generality that no two chords share an endpoint. Number the endpoints of the chords from 1 to $2N$ in the order as they appear as we move clockwise around the circle (from an arbitrary but fixed starting point).

The idea is simple. For $1 \leq i < j \leq 2N$, let $G(i,j)$ denote the subgraph of $G$ induced by chords whose both endpoints belong to the set $\{i, i+1, \ldots, j\}$. Obviously $G = G(1, 2N)$.

Let $1 \leq i < j \leq 2N$. If $j = i+1$ then the solution to EWIS$(G(i,j), w, k)$ is *yes* if and only if either $k = 0$, or $(i, i+1)$ is a chord and $k = w((i, i+1))$.

Otherwise, let $r$ be the other endpoint of the chord whose one endpoint is $j$. If $r < i$ or $r > j$, then no independent set of the graph $G(i,j)$ contains the chord $(r,j)$, so the solution to EWIS$(G(i,j), w, k)$ is *yes* if and only if the solution to EWIS$(G(i,j-1), w, k)$ is *yes*. Suppose now that $i \leq r \leq j-1$ and let $I$ be an independent set of $G(i,j)$. The set $I$ may or may not contain the chord $(r,j)$. If $I$ does not contain $(r,j)$, then $I$ is an independent set of of $G(i,j-1)$ as well. If $I$ contains $(r,j)$, then no other chord in $I$ can intersect the chord $(r,j)$. In particular, this implies that $I$ is of the form $I = \{(r,j)\} \cup I_1 \cup I_2$ where $I_1$ is an independent set of $G(i,r-1)$ and $I_2$ is an independent set of $G(r+1, j-1)$.

Therefore, the solution to EWIS$(G(i,j), w, k)$ is *yes* if and only if either the solution to EWIS$(G(i,j-1), w, k)$ is *yes*, or the solution to EWIS$(G', w, k)$ is *yes*, where $G'$ is the graph whose connected components are $G[\{(r,j)\}]$, $G(i,r-1)$ and $G(r+1,j-1)$. Assuming that the solutions for $G(i,r-1)$ and $G(r+1,j-1)$ have already been obtained recursively, we can apply Corollary 7 in this case.

The above discussion implies an obvious $O(M^2n^2)$ algorithm that correctly solves the problem. □

### Graphs of clique-width at most $k$

The *clique-width* of a graph $G$ is defined as the minimum number of labels needed to construct $G$, using the following four graph operations:

(*i*) Create a new vertex $v$ with label $i$ (denoted by $i(v)$).

(*ii*) Take the disjoint union of two labeled graphs $G$ and $H$ (denoted by $G \oplus H$).

(*iii*) Join by an edge each vertex with label $i$ to each vertex with label $j$ ($i \neq j$, denoted by $\eta_{i,j}$).

(*iv*) Rename label $i$ to $j$ (denoted by $\rho_{i \to j}$).

An expression built from the above four operations is called a *clique-width expression*. A clique-width expression using $k$ labels is called a *$k$-expression*. Each $k$-expression $t$ uniquely defines a labeled graph $lab(t)$, where the labels are integers $\{1, \ldots, k\}$ associated with the vertices and each vertex has exactly one label. We say that a $k$-expression $t$ defines a graph $G$ if $G$ is equal to the graph obtained from the labeled graph $lab(t)$ after removing its labels. The clique-width of a graph $G$ is equal to the minimum $k$ such that there exists a $k$-expression defining $G$.

Many graph problems that are **NP**-hard for general graphs are solvable in linear time when restricted to graphs of clique-width at most $k$, if a $k$-expression is given as part of the input [13]. EWIS is no exception.

**Theorem 22.** *For every fixed* $k$, EWIS *admits an* $O(M^2l)$ *algorithm for graphs of clique-width at most* $k$*, where* $l$ *is the number of operations in a given* $k$*-expression for* $G$*.*

*Proof.* Suppose that the labels are integers $\{1, \ldots, k\} = [k]$. For every subset of labels $S \subseteq [k]$, let $\text{EWIS}(G, w, S, m)$ denote the answer to the following question: "Is there an independent set of $G$ with total weight $m$ that contains exactly the labels from $S$?"

Given a $k$-expression $t$ defining the input graph $G$, we can solve $\text{EWIS}(G, w, M)$ by first computing all the values for $\text{EWIS}(G, w, S, m)$, for every subset of labels $S \subseteq [k]$, and every $m \in [M]$. It is easy to see that this can be performed in time $O(M^2 l)$ by the following dynamic programming algorithm.

If $|V| = 1$ then let $v \in V$. For all $S \subseteq [k]$, and for all $m \in [M]$, let

$$\text{EWIS}(G, w, S, m) = \begin{cases} yes, & \text{if } S = \{label(v)\} \text{ and } m = w(v); \\ no, & \text{otherwise.} \end{cases}$$

If $G = G_1 \oplus G_2$ then let for all $S \subseteq [k]$, and for all $m \in [M]$:

$$\text{EWIS}(G, w, S, m) = \begin{cases} yes, & \text{if } \text{EWIS}(G_1, w, S, m) = yes; \\ yes, & \text{if } \text{EWIS}(G_2, w, S, m) = yes; \\ yes, & \text{if there is an } m' \in [m-1] \text{ such that} \\ & \quad \text{EWIS}(G_1, w, S, m') = \text{EWIS}(G_2, w, S, m - m') = yes; \\ no, & \text{otherwise.} \end{cases}$$

This can be computed in time $O(M^2)$, similarly as in Corollary 7.
If $G = \eta_{i,j}(G_1)$ then let for all $S \subseteq [k]$, and for all $m \in [M]$:

$$\text{EWIS}(G, w, S, m) = \begin{cases} \text{EWIS}(G_1, w, S, m), & \text{if } \{i, j\} \not\subseteq S; \\ no, & \text{otherwise.} \end{cases}$$

If $G = \rho_{i \to j}(G_1)$ then let for all $S \subseteq [k]$, and for all $m \in [M]$:

$$\text{EWIS}(G, w, S, m) = \begin{cases} \text{EWIS}(G_1, w, S, m), & \text{if } S \cap \{i, j\} = \emptyset; \\ \text{EWIS}(G_1, w, S \cup \{i\}, m), & \text{if } S \cap \{i, j\} = \{j\}; \\ no, & \text{otherwise.} \end{cases}$$

Having computed all the values $\text{EWIS}(G, w, S, m)$, the solution to $\text{EWIS}(G, w, M)$ is clearly given by

$$\text{EWIS}(G, w, M) = \begin{cases} yes, & \text{if there is an } S \subseteq [k] \text{ such that } \text{EWIS}(G, w, S, M) = yes; \\ no, & \text{otherwise.} \end{cases}$$

□

### Chordal graphs

**Theorem 23.** EWIS *admits an* $O(M^2 n(n+m))$ *algorithm for chordal graphs.*

*Proof.* Given a chordal graph $G$, we first compute a *clique tree* of $G$. This can be done in time $O(n+m)$ [24]. A clique tree of a chordal graph $G$ is a tree $T$ whose nodes are the maximal cliques of $G$, such that for every vertex $v$ of $G$, the subgraph $T_v$ of $T$ induced by the maximal cliques containing $v$ is a tree. Furthermore, we fix an arbitrary node $K_r$ in the clique tree in order to obtain a *rooted clique tree*. For a maximal clique $K$, we denote by $G(K)$ the subgraph of $G$ induced by the vertices of $K$ and all vertices contained in some descendant of $K$ in $T$.

The algorithm is based on a set of identities developed by Okamoto *et al.* in [31], where a clique tree representation was used to develop linear-time algorithms to count independent sets in a chordal graph. Let $\mathcal{IS}(G)$ be the family of independent sets in $G$. For a vertex $v$, let $\mathcal{IS}(G,v)$ be the family of independent sets in $G$ that contain $v$. For a vertex set $U$, let $\overline{\mathcal{IS}}(G,U)$ be the family of independent sets in $G$ that contain no vertex of $U$. Consider a maximal clique $K$ of $G$, and let $K_1,\ldots,K_l$ be the children of $K$ in $T$. (If $K$ is a leaf of the clique tree, we set $l=0$.) Then, as shown in [31], for every distinct $i,j\in[l]$, the sets $V(G(K_i))\backslash K$ and $V(G(K_j))\backslash K$ are disjoint. Moreover, if $\sqcup$ denotes the disjoint union, the following relations hold:

$$\begin{aligned}
\mathcal{IS}(G(K)) &= \overline{\mathcal{IS}}(G(K),K)\sqcup\bigsqcup_{v\in K}\mathcal{IS}(G(K),v)\,;\\
\mathcal{IS}(G(K),v) &= \left\{I\cup\{v\}\;\middle|\; I=\bigcup_{i=1}^{l}I_i,\ I_i\in\begin{cases}\mathcal{IS}(G(K_i),v), & \text{if } v\in K_i;\\ \overline{\mathcal{IS}}(G(K_i),K\cap K_i), & \text{otherwise.}\end{cases}\right\};\\
\overline{\mathcal{IS}}(G(K),K) &= \left\{I\;\middle|\; I=\bigsqcup_{i=1}^{l}I_i,\ I_i\in\overline{\mathcal{IS}}(G(K_i),K\cap K_i)\right\};\\
\overline{\mathcal{IS}}(G(K_i),K\cap K_i) &= \overline{\mathcal{IS}}(G(K_i),K_i))\sqcup\bigsqcup_{u\in K_i\backslash K}\mathcal{IS}(G(K_i),u)\quad\text{for each } i\in[l]\,.
\end{aligned}$$

We extend our usual Boolean predicate $\mathrm{EWIS}(H,w,k)$ to the following two: for a vertex $v$ of a weighted graph $(H,w)$ and in integer $k$, let $\mathrm{EWIS}(H,w,k,v)$ denote the Boolean predicate that is *yes* if and only if in $H$ there is an independent set $I$ of total weight $k$ that contains $v$. Also, for a set of vertices $U$ let $\overline{\mathrm{EWIS}}(H,w,k,U)$ take the value *yes* if and only if in $H$ there is an independent set of total weight $k$ that contains no vertex from $U$. Based on the above equations, we can develop the following recursive relations for EWIS:

$$\mathrm{EWIS}(G(K),w,k)=\overline{\mathrm{EWIS}}(G(K),w,k,K)\vee\bigvee_{v\in K:w(v)\le k}\mathrm{EWIS}(G(K),w,k,v)\tag{3}$$

where $\vee$ denotes the usual *Boolean OR function* (with the obvious identification *yes* $\leftrightarrow 1$, *no* $\leftrightarrow 0$). That is, its value is *yes* if at least one of its arguments is *yes*, and *no* otherwise.

$$\mathrm{EWIS}(G(K),w,k,v)=\mathbf{GSS}(A_1,\ldots,A_l,k-w(v))\tag{4}$$

where $\mathbf{GSS}(A_1,\ldots,A_l,k)$ denotes the solution to the generalized subset sum problem with the input $(A_1,\ldots,A_l,k)$, and the sets $A_i$ for $i\in[l]$ are given by

$$A_i=\begin{cases}\{k'-w(v)\ :\ w(v)\le k'\le k,\ \mathrm{EWIS}(G(K_i),w,k',v)=yes\}, & \text{if } v\in K_i;\\ \{k'\ :\ 1\le k'\le k,\ \overline{\mathrm{EWIS}}(G(K_i),w,k',K\cap K_i)=yes\}, & \text{otherwise.}\end{cases}$$

Note that if $I_i\in\mathcal{IS}(G(K_i),v)$ and $I_j\in\mathcal{IS}(G(K_j),v)$ for some distinct indices $i,j\in[l]$, then we have $I_i\cap I_j=\{v\}$. Moreover, since this is the only possible nonempty intersection of two independent sets from $\bigcup_{i=1}^{l}I_i$ in the equation for $\mathcal{IS}(G(K),v)$, it follows that the sum of the weights of the sets $I_i\backslash\{v\}$ (over all $i\in[l]$) equals to the weight of $\left(\bigcup_{i=1}^{l}I_i\right)\backslash\{v\}$, thus justifying Equation (4).

Similarly, we have

$$\overline{\mathrm{EWIS}}(G(K),w,k,K)=\mathbf{GSS}(A_1,\ldots,A_l,k)\tag{5}$$

where, for each $i\in[l]$, the set $A_i$ is given by

$$A_i=\{k':1\le k'\le k,\ \overline{\mathrm{EWIS}}(G(K_i),w,k',K\cap K_i)=yes\}\,,$$

and, finally, for each $i \in [l]$, we have:

$$\overline{\text{EWIS}}(G(K_i), w, k, K \cap K_i) = \overline{\text{EWIS}}(G(K_i, w, k, K_i)) \vee \bigvee_{u \in K_i \setminus K} \text{EWIS}(G(K_i), w, k, u)\,. \quad (6)$$

Given the above equations, it is now easy to develop a pseudo-polynomial dynamic programming algorithm. Having constructed a rooted tree $T$ of $G$, we traverse it in a bottom-up manner. For a leaf $K$, we have

$$\overline{\text{EWIS}}(G(K), w, k, K) = \begin{cases} yes, & \text{if } k = 0; \\ no, & \text{otherwise.} \end{cases}$$

and

$$\text{EWIS}(G(K), w, k, v) = \begin{cases} yes, & \text{if } w(v) = k; \\ no, & \text{otherwise.} \end{cases} .$$

For every other node $K$, we compute the values of $\overline{\text{EWIS}}(G(K), w, k, K)$ and $\text{EWIS}(G(K), w, k, v)$ by referring to the recursive relations (6), (5) and (4) in this order. Finally, the value of $\text{EWIS}(G, w, k)$ is given by $\text{EWIS}(G(K_r), w, k)$, which can be computed using Equation (3).

The correctness of the procedure follows immediately from the above discussion. To justify the time complexity, observe that in a node $K$ of the tree with children $K_1, \ldots, K_l$, the number of operations performed is $O(\sum_{i=1}^{l} |K_i| + lM^2 + |K|lM^2)$. Summing up over all the nodes of the clique tree, and using the fact that a chordal graph has at most $n$ maximal cliques, which satisfy $\sum_{K \in V(T)} |K| = O(n + m)$, the claimed complexity bound follows. □

### $mK_2$-free graphs

**Theorem 24.** *For every positive integer $m$,* EWIS *admits a pseudo-polynomial algorithm for $mK_2$-free graphs.*

*Proof.* All maximal independent sets $I_1, \ldots, I_N$ in an $mK_2$-free graph can be found in polynomial time [1, 3, 33, 38]. Since every independent set is contained in a maximal one, $\text{EWIS}(G, w, k)$ will take the value *yes* if and only if there is an $i \in [N]$ such that $\text{EWIS}(G[I_i], w, k)$ is *yes*. Thus, the EWIS problem in $mK_2$-free graphs reduces to solving polynomially many instances of the subset sum problem. □

### Subclasses of $P_5$- and fork-free graphs

**Theorem 10.** EWIS *is solvable in pseudo-polynomial time in each of the following classes: ($P_5$, double-gem, co-domino)-free graphs (and their subclass ($P_5$, co-P)-free graphs), (bull, fork)-free graphs, (co-P, fork)-free graphs, and ($P_5$, fork)-free graphs.*

*Proof.* This theorem essentially follows from Theorem 9 and the results in [10] and [23] (see also [9] for some applications of modular decomposition to the WIS problem). We briefly summarize the main ideas.

Every prime ($P_5$, *double-gem, co-domino*)-free graph is $2K_2$-free (the complementary version of this statement is proved in [23]). Since we can easily extend Theorem 24 to the extended version of EWIS, this implies the result for ($P_5$, *double-gem, co-domino*)-free graphs.

The (extended) EWIS problem can be solved in pseudo-polynomial time for *co-gem*-free graphs. Indeed, for every vertex $v$ of a *co-gem*-free graph $G$, the non-neighborhood of $v$ in $G$ is $P_4$-free. So the problem reduces to solving $O(nM)$ subproblems in $P_4$-free graphs, which can

be done for example by modular decomposition. Every $P_4$-free graph is either disconnected, or its complement is disconnected. Thus, the only prime $P_4$-free graph is the graph on a single vertex.

In [10], it is shown that prime graphs that contain a *co-gem* and are either $(bull, fork)$-free, $(co\text{-}P, fork)$-free or $(P_5, fork)$-free have a very simple structure. The (extended) EWIS problem can be solved in pseudo-polynomial time for such graphs. Together with the above observation about *co-gem*-free graphs and Theorem 9, this concludes the proof. □

## Exact problems for graphs of bounded clique-width

As shown by Courcelle *et al.* [13], every problem expressible in monadic second order logic with quantification over the vertices and vertex sets can be solved in linear time for graphs of clique-width at most $k$. More formally, using the terminology from [13]:

**Theorem 25** (Theorem 4 from [13])**.** *Let $\mathcal{C}$ be a class of p-graphs of clique-width at most $k$, each given by a $k$-expression $g$ defining it. Then every LinEMSOL($\tau_{1,p}$) optimization problem over $\mathcal{C}$ can be solved in time $O(l(g))$, where $l(g)$ is the number of operations in $g$. A corresponding algorithm can be efficiently constructed from the logical formula describing the problem.*

A simple modification of the original proof of this theorem from [13] shows that a similar conclusion holds true for *LinEMSOL($\tau_{1,p}$) exact problems over $\mathcal{C}$.*

**Definition 2** (*LinEMSOL($\tau$)* Exact Problems over $K$)**.** *For a vocabulary $\tau$ and a class of $\tau$-structures $K$, we say that a decision problem $P$ is a LinEMSOL($\tau$) exact problem over $K$ if it can be expressed in the following form:*
**Instance:** *A $\tau$-structure $\mathcal{A} \in K$, $m$ evaluation functions $f_1, \ldots, f_m$ associating non-negative integer values to the elements of $\mathcal{A}$, and a non-negative integer $b$.*
**Question:** *Is there an assignment $z$ to the free variables in $\theta$ such that*

$$\langle \mathcal{A}, z \rangle \models \theta(X_1, \ldots, X_l) \quad \text{and} \quad \sum_{i \in [l], j \in [m]} a_{ij} |z(X_i)|_j = b\,,$$

*where $\theta$ is an $MSOL(\tau)$ formula having free set variables $X_1, \ldots, X_l$, $|z(X_i)|_j$ is a short notation for $\sum_{a \in z(X_i)} f_j(a)$, and $\{a_{ij} : i \in [l], j \in [m]\}$ are non-negative integers.*

**Theorem 26.** *Let $\mathcal{C}$ be a class of p-graphs of clique-width at most $k$, each given by a $k$-expression $g$ defining it. Then every LinEMSOL($\tau_{1,p}$) exact problem over $\mathcal{C}$ can be solved in time $O(b^2 l(g))$, where $l(g)$ is the number of operations in $g$. A corresponding algorithm can be efficiently constructed from the logical formula describing the problem.*

*Proof.* Let $[b]_0 = \{0, 1, \ldots, b\}$ denote the set of non-negative integers not exceeding $b$.

We prove Theorem 26 by mimicking the proof of Theorem 4 from [13] (pages 144–146), with the following modification:

For $A \subseteq \mathcal{P}(V(G))^l$, instead of defining $Max\_h(A)$, define

$$h(A) = \{h(D_1, \ldots, D_l) : (D_1, \ldots, D_l) \in A\} \cap [b]_0\,.$$

Items (1) and (2) from [13] then become, respectively:

$$h(A \boxtimes B) = \{h(D_1, \ldots, D_n) + h(D'_1, \ldots, D'_n) : (D_1, \ldots, D_n) \in A, (D'_1, \ldots, D'_n) \in B\} \cap [b]_0 \quad (7)$$

and

$$h(A \cup B) = h(A) \cup h(B)\,. \quad (8)$$

Also, from the definition of $LinEMSOL(\tau)$ exact problems it follows that a $LinEMSOL(\tau_{1,p})$ exact problem over a class of graphs $K$ can be formulated as verifying whether $b \in h(sat(G, \theta))$ for a given graph $G \in K$ presented over $\tau_{1,p}$, where $\theta$ is a fixed $MSOL(\tau_{1,p})$ formula.

Part $(i)$ of the proof remains unchanged. Part $(ii)$ of the proof becomes:

$(ii)$ Traverse $Tree(g)$ from bottom to top and, at each node $x$ and for each formula $\varphi$ assigned to $x$ by the previous step, compute $h(sat(Graph(x), \varphi))$ as follows:

- If $x$ is a leaf compute $h(sat(Graph(x), \varphi))$ directly.
- If $x$ corresponds to a unary operation, set $h(sat(Graph(x), \varphi)) = h(sat(Graph(y), \varphi'))$ where $y$ is the son of $x$, and $\varphi'$ is the formula assigned to $y$ by the previous step.
- If $x$ corresponds to the binary operation $\oplus$ then using (7)–(8) compute $h(sat(Graph(x), \varphi))$ from the two lists of values: $h(sat(Graph(u), \varphi'_j))$ and $h(sat(Graph(v), \psi'_j))$, for $j \in [m]$, where $u$ and $v$ are the sons of $x$ in $Tree(g)$ and $\varphi'_j$ and $\psi'_j$ are the lists of formulas assigned to $u$ and $v$ by the previous step, respectively.

Also at each node $x$, each formula $\varphi$ assigned to $x$ and each $b' \in h(sat(Graph(x), \varphi))$, keep one tuple of $sat(Graph(x), \varphi)$ that $h$ maps to $b'$.

Finally, the claimed time complexity $O(b^2 l(g))$ follows from the fact that in each of the $O(l(g))$ nodes of $Tree(g)$, we perform constantly many computations of the type (7) or (8), each of which takes $O(b^2)$ and $O(b)$ time, respectively. □