On Dynamic Threshold Graphs and Related Classes ☆

Tiziana Calamoneri, Angelo Monti, Rossella Petreschi

[a]Computer Science Department,
"Sapienza" University of Rome, Italy
calamo@di.uniroma1.it, monti@di.uniroma1.it, petreschi@di.uniroma1.it

Abstract

This paper deals with the well known classes of threshold and difference graphs, both characterized by *separators*, i.e. node weight functions and thresholds. We design an efficient algorithm to find the minimum separator, and we show how to maintain minimum its value when the input (threshold or difference) graph is fully dynamic, i.e. edges/nodes are inserted/removed. Moreover, exploiting the data structure used for maintaining the minimality of the separator, we study the disjoint union and the join of two threshold graphs, showing that the resulting graphs are threshold signed graphs, i.e. a superclass of both threshold and difference graphs. Finally, we consider the complement operation on all the three introduced classes of graphs.

All these operations produce in output the modified graph in terms of their separator and require time linear w.r.t. the number of different degrees. We observe that recomputing from scratch the separator would run either in linear (for threshold and difference graphs) or quadratic (for threshold signed graphs) time w.r.t. the number of nodes of the graph.

Keywords: fully dynamic graphs, threshold graphs, difference graphs, chain graphs, threshold signed graphs, graph operations.

☆Partially supported by the Italian Ministry of Education and University, PRIN project "AMANDA: Algorithmics for MAssive and Networked DAta" and by Sapienza University of Rome.

1. Introduction

In many applications of graph algorithms, *graphs* are *fully dynamic*, i.e. both edges and nodes may be inserted or eliminated.

Typically, one would like to answer to a precise query on the fully dynamic graph, so the goal is to update the data structure after dynamic changes, rather than to recompute it from scratch each time.

In this paper we deal with the maintenance of fully dynamic graphs when restricted to the classes of threshold and difference graphs.

Threshold graphs were introduced in 1977, independently as a model for three different problems: aggregation of inequalities in integer programming for set packing problems [7], node-labeling of the graph associated to a scheduling problem [11] and synchronization in parallel processing, lockout- and deadlock-free, solved with a semaphore-based approach [13]. After that, they have been defined many other times, as they are a natural model for a number of problems in many fields, e.g. resource allocation problems [20], scheduling [14], efficient parallel joins in relational databases [17], polyhedral combinatorics [5, 4] and spectral graph theory [1].

The first definition of *difference graphs* –also known as *chain graphs*– goes back to 1972 [15] but they have been re-discovered in 1982 independently during an in depth analysis of threshold graphs [8] and to model a problem on partial orders [23]. Even these graphs have been then exploited in many fields such as recognizing poset dimension [23], studying the learning behavior of children [10], distributed memory multiprocessors [2], modeling software development process [9], dynamic networks [18] and cytoplasmic incompatibility in biology [19].

The pervasive application of threshold and difference graphs in so many fields makes natural to handle them in a fully dynamic way. To the best of our knowledge, few works deal with this topic. Namely, in [22] the problem of dynamically recognizing some classes of graphs (and among them threshold graphs) is handled. In [12] the authors consider the problem of adding/deleting edges with the aim of transforming a given graph into a threshold graph with the minimum number of changes. This paper is a contribution to the problem of the dynamic maintenance of threshold and difference graphs.

Among the numerous equivalent definitions of threshold and difference graphs, many exploit a node weight function and a threshold. This pair

is called a *separator* and it is not unique. It is of interest to determine a *minimum separator*, i.e. a separator with minimum value of the threshold.

In this paper we present a new algorithm for finding a minimum separator. This algorithm, interesting by itself for its simplicity and linearity, is then considered as a pre-computation for maintaining the minimality after fully dynamically changing the input graph. To do this, we propose a simple data structure for maintaining the minimality of the separator, and handle some binary operations of two threshold graphs (disjoint union and join) whose result is in general not in the same graph class anymore, but in a superclass, called threshold signed graphs [3]; this superclass can be defined in terms of a node weight function and of two thresholds.

Finally, we tackle the problem of computing the weight function and the thresholds of the complement of one of these graphs without recomputing them from scratch, taking into account that the classes of threshold and threshold signed graphs are closed under complement while the complement of a difference graph is a threshold signed graph. To the best of our knowledge, it is the first time that the weight and thresholds of the complement of a given threshold signed graph is directly computed.

All the operations presented in this paper run in linear time w.r.t. the number of different degrees in the graph. This is particularly important, because recomputing from scratch the node weight function would require either linear (for threshold and difference graphs) or quadratic (for threshold signed graphs) time w.r.t. the number of nodes of the graph.

The paper is organized as follows: in Section 2 we recall some definitions. In Section 3, we describe the assignment algorithm determining a minimal integral separator for threshold and difference graphs. Moreover, we describe the data structures we use to store threshold and difference graphs; thanks to them, we are able to guarantee that all the operations handled in the next sections work in time that is linear w.r.t. the number of different degrees in the graph. Sections 4 and 5 describe how to add/delete an edge or a node, respectively, in either a threshold or a difference graph whenever it is possible to result in a graph of the same class (threshold or difference graphs). Section 6 describes a data structure feasible to store threshold signed graphs. In Section 7 we consider the operations of disjoint union and join of two threshold graphs and we show that the result is a threshold signed graph. In Section 8 we address the problem of adjusting the node weight function and the threshold(s) when the complement operation is applied

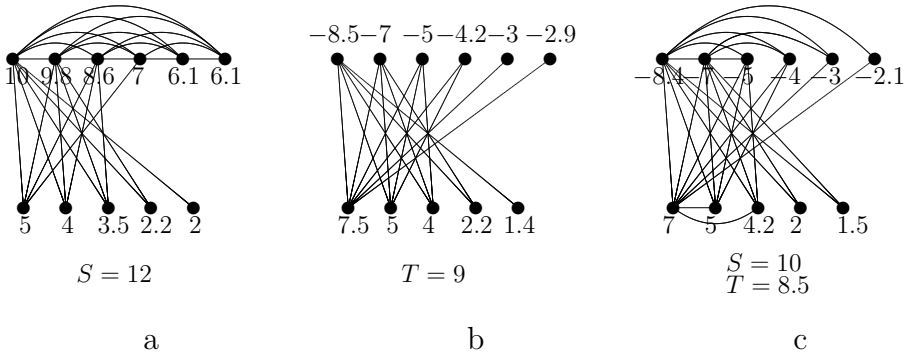

Figure 1: a. A threshold graph; b. A difference graph; c. A threshold signed graph.

to threshold, difference and threshold signed graphs, respectively. Finally, Section 9 concludes the paper with some observations and open problems.

2. Preliminaries

In this section, we list some definitions and properties useful for the rest of the paper. All the references of the results listed in this section can be found in the comprehensive survey book on threshold graphs, difference graphs and related topics by Mahadev and Peled [16].

Given a graph $G = (V, E)$, a subset of nodes $V' \subseteq V$ induces a *stable set* if $\forall\, u, v \in V'$ edge $(u, v) \notin E$; vice-versa, $V' \subseteq V$ induces a *clique* if $\forall\, u, v \in V'$ edge $(u, v) \in E$.

We denote by $\deg(v)$ the *degree* of node $v \in V$. A node v is *isolated* if $\deg(v) = 0$.

As we have already pointed out, there are many equivalent definitions of threshold graphs; in this paper we use the following:

Definition 1. *A graph $G = (V, E)$ is a* threshold graph *if there is a mapping $a : V \to \mathbb{R}^+$ and a positive real number S such that*

$$a(v) < S \text{ for all } v \in V \tag{1}$$

$$(v, w) \in E \text{ if and only if } a(v) + a(w) \geq S \tag{2}$$

The pair (a, S) will be called separator *for graph G.*

In Figure 1.a a threshold graph with one of its separators is depicted.

Informally speaking, the nodes of a threshold graph can be partitioned into two sets, one inducing a clique and one a stable set; these sets are connected by a difference graph, that can be defined as follows:

Definition 2. *A graph $G = (V, E)$ is a* difference graph *if there is a mapping $a : V \to \mathbb{R}$ and a positive real number T such that*

$$|a(v)| < T \text{ for all } v \in V \tag{3}$$

$$(v, w) \in E \text{ if and only if } |a(v) - a(w)| \geq T \tag{4}$$

The pair (a, T) will be called separator *for graph G.*

The node set of a difference graph G can be partitioned as $V = U \cup W$, where $U = \{v \in V : a(v) \geq 0\}$ and $W = \{v \in V : a(v) < 0\}$; both U and W induce a stable set and hence G is bipartite with bipartition (U, W). A difference graph with one of its separators is shown in Figure 1.b.

Although Definitions 1 and 2 are very similar, the two defined classes are incomparable (indeed, they have not empty intersection – e.g. a star $K_{1,n-1}$ is both a difference and a threshold graph – but they are not included one into the other one). Nevertheless they are strictly related, as shown by the following theorem:

Theorem 3. *A bipartite graph $G = (U \cup W, E)$ is a difference graph if and only if adding to G all possible edges with both ends in U yields a threshold graph.*

The nodes of both threshold and difference graphs can be partitioned into two sets, in such a way that the neighborhoods of the nodes in each set are included one into the other one and form either one or two chains of inclusions (see e.g. Figures 1.a and 1.b).

Now we introduce the notion of *degree partition*.

Definition 4. *Let $G = (V, E)$ be a graph whose distinct node-degrees are $\delta_1 < \ldots < \delta_m$, and let $\delta_0 = 0$ (even if no node of degree 0 exists). Let $D_i = \{v \in V \text{ s.t. } deg(v) = \delta_i\}$ for $i = 0, \ldots, m$; D_i is called i-*th box*; the sequence $D_0, \ldots, D_m$ is called the* degree partition *of G.*

We naturally extend the previous definition to bipartite graphs:

Definition 5. *Let $G = (U \cup V, E)$ be a bipartite graph whose distinct node-degrees for the nodes in the partition $X \in \{U, W\}$ are $\delta_1^X < \ldots < \delta_{t_X}^X$ and let $\delta_0^X = 0$ (even if no node of degree 0 exists in the partition X). Let $D_i^X = \{v \in X \text{ s.t. } deg(v) = \delta_i^X\}$ for $0 \leq i \leq t_X$; the sequence $D_0^U, \ldots, D_{t_U}^U, D_0^W, \ldots, D_{t_W}^W$ is called the* bipartite degree partition *of G.*

The following lemmas show that the notion of degree partition (bipartite degree partition) is crucial for understanding threshold (difference) graphs.

Lemma 6. *Let $G = (V, E)$ be a threshold graph with degree partition $D_0, \ldots, D_m$ and whose node set is partitioned into a clique K and a stable set I; let be given two distinct nodes $x \in D_i$ and $y \in D_j$ for some i and j.*

1. *$D_0 \cup \ldots \cup D_{\lfloor m/2 \rfloor} = I$ and $D_{\lfloor m/2 \rfloor + 1} \cup \ldots \cup D_m = K$;*
2. *If $e = (x, y) \notin E$, the graph $G' = (V, E \cup \{e\})$ is a threshold graph if and only if $i + j = m$;*
3. *If $e = (x, y) \in E$, the graph $G' = (V, E \setminus \{e\})$ is a threshold graph if and only if $i + j = m + 1$;*
4. *$e = (x, y) \in E$ if and only if $i + j \geq m + 1$.*

Lemma 7. *Let $G = (U \cup W, E)$ be a difference graph with degree partition $D_0^U, \ldots, D_{t_U}^U, D_0^W, \ldots, D_{t_W}^W$, and let be given two distinct nodes $x \in D_i^U$ and $y \in D_j^W$ for some i and j.*

1. *$t_U = t_W = t$;*
2. *If $e = (x, y) \notin E$, the graph $G' = (U \cup W, E \cup \{e\})$ is a difference graph if and only if $i + j = t$;*
3. *If $e = (x, y) \in E$, the graph $G' = (U \cup W, E \setminus \{e\})$ is a difference graph if and only if $i + j = t + 1$;*
4. *$e = (x, y) \in E$ if and only if $i + j \geq t + 1$.*

Notice that threshold (difference) graphs are univocally determined by their degree (bipartite degree) partition.

We now introduce a superclass of both threshold and difference graphs.

Definition 8. *A graph $G = (V, E)$ is a* threshold signed graph *if there is a mapping $a : V \to \mathbb{R}$ and two positive real numbers S and T such that*

$$|a(v)| < min\{S, T\} \tag{5}$$

$$(v, w) \in E \text{ iff either } |a(v) + a(w)| \geq S \text{ or } |a(v) - a(w)| \geq T. \tag{6}$$

The triple (a, S, T) will be called separator *for graph G.*

Consider $X = \{x \in V \text{ s.t. } a(x) < 0\}$ and $Y = \{x \in V \text{ s.t. } a(x) \geq 0\}$. As highlighted in Figure 1.c, we can see a threshold signed graph as constituted by two threshold graphs, G^- and G^+ respectively induced by X and Y, that are connected by a difference graph D. Notice that for X we consider the opposite of the a's values.

3. A data structure for computing minimal integral separator for threshold or difference graphs

Although in Definition 1 a threshold graph $G = (V, E)$ is a graph having a separator with non-negative *real* values, it is common to equivalently require the separator to have non-negative *integral* values (i.e. an integral separator) [16].

We say that an integral separator (a, S) for G is *minimum* if for any other integral separator (a', S') for G we have $S \leq S'$. In the following theorem we show that the value of S of a minimum integral separator (a, S) of G is given by the dimension m of the degree partition of G plus 1.

Theorem 9. *Let $G = (V, E)$ be a threshold graph with degree partition $D_0, \ldots, D_m$. The pair (a, S), where $S = m + 1$ and for each node $v \in V$, $a(v) = i$ if $v \in D_i$, is a minimal integral separator of G.*

Proof. First of all, we prove that (a, S) is a separator, i.e. that satisfies the two inequalities of Definition 1. Note that for each $v \in V$ it holds $0 \leq a(v) \leq m < S$ thus the pair (a, S) satisfies Inequality 1. Moreover, Inequality 2 follows from Item 4 in Lemma 6. Trivially, (a, S) is integral.

Let us now prove that (a, S) is minimal. By contradiction, let (a, S) be not minimal, and let (a', S') be an integral separator for G such that $S' < S$. Observe that only isolated nodes can have weight equal to zero (indeed, if $(u, v) \in E$ and $a'(u) = 0$ then $a'(u) + a'(v) = a'(v) < S'$ from Inequality 1 of Definition 1, but this contradicts Inequality 2 of Definition 1).

Moreover, notice that two nodes u and v having the same weight necessarily behave in the same way (i.e. for any other node $w \in V$, it holds that $(u, w) \in E$ if and only if $(v, w) \in E$), so nodes having different degree cannot have the same weight.

All this implies that the function a' on the non isolated nodes assume at least m different strictly positive weights. Thus Inequality 1 of Definition 1 implies that $S' \geq m + 1$. The chain $m + 1 = S > S' \geq m + 1$ proves the minimality of (a, S). ∎

From the same reasonings as in the proof of Theorem 9, we deduce the following assignment for the minimal integral separator (a, T) of a difference graph G.

Theorem 10. *Let $G = (U \cup W, E)$ be a difference graph with bipartite degree partition $D_1^U, \ldots, D_t^U$, $D_1^W, \ldots, D_t^W$. The pair (a, S), where $S = 2t + 1$ and, for each node $v \in U$, $a(v) = -i$ if $v \in D_i^U$ and $a(v) = i$ if $v \in D_i^W$, is a minimal integral separator of G.*

Notice that Orlin [21] and later Ordman [20] show how to minimize the separator for threshold graphs considering an equivalent definition requiring that the sum of the weights of the nodes of any independent set has to be smaller than the threshold. The value of this threshold is larger than the value of S computed in this paper.

Now we present two data structures for representing threshold and difference graphs allowing us to compute in a natural way minimal integral separators for these graph classes, according to Theorems 9 and 10. Since a threshold graph $G = (V, E)$ is univocally determined by its degree partition $D_0, \ldots, D_m$, we store G using two arrays $\delta[0..m]$ and $\mu[0..m]$, where $\delta[i]$ represents the degree δ_i of the nodes in the i-th box D_i, and $\mu[i]$ represents its cardinality, $|D_i|$. Similarly, let $G = (U \cup W, E)$ be a difference graph with bipartite degree partition $D_1^U, \ldots, D_t^U$, $D_1^W, \ldots, D_t^W$. This partition univocally determines G, and G may be represented using the arrays $\delta_X[0..t]$ and $\mu_X[0..t]$, where $\delta_X[i]$ represents the degree of the nodes in the i-th box D_i^X, and $\mu_X[i]$ represents its cardinality $|D_i^X|$, $X \in \{U, W\}$. If $G = (V, E)$ is a threshold graph, i is the weight associated to all the $\mu[i]$ nodes belonging to the box D_i of degree δ_i. Similarly, if $G = (U \cup W, E)$ is a difference graph, i is the weight associated to all the $\mu_X[i]$ nodes belonging to box D_i^X of degree $\delta_X[i]$, with $X \in \{U, W\}$.

Exploiting these two data structures, the following theorem holds:

Theorem 11. *Given a threshold (difference) graph G by means of its (bipartite) degree partition, its minimal integral separator can be found in time linear w.r.t. the number of different degrees in G.*

In the next two sections we will exploit the data structures introduced here in order to efficiently manipulate threshold and difference graphs, so ensuring that the integral separator remains minimal even for the graphs resulting from the operations handled in the next sections.

There, each time we speak about a graph G (either threshold or difference), G is represented by means of the arrays δ and μ.

4. Adding/deleting an edge to threshold/difference graphs

In this section we study how to get a new graph, obtained by adding/deleting an edge from a graph that is either a threshold or a difference graph, and to keep immediately available the knowledge of the minimum separator for the new graph.

In order to make easier the exposition, preliminarily we consider two functions, operating on the data structures introduced in Section 3.

By **IncreaseDeg**(δ, μ, i, dim) we denote the operation of updating arrays $\delta[0..dim]$ and $\mu[0..dim]$ when the degree of a node in box D_i, $0 \leq i \leq dim$, is increased by one.

IncreaseDeg can have as consequence the appearance of a new box (if $i = dim$ or if the degree of the nodes in D_{i+1} is different from the degree of nodes in D_i plus one). On the other hand, this increment can also have as consequence the disappearance of the box D_i (if $i \neq 0$, $|D_i| = 1$ and the degree of the nodes in D_{i+1} is equal to the degree of the nodes in D_i plus one).

Symmetrically, we may consider the operation **DecreaseDeg**(δ, μ, i, dim) of updating arrays $\delta[0..dim]$ and $\mu[0..dim]$ when the degree of a node in box D_i, $0 < i \leq dim$, is decreased by one.

The execution of both **IncreaseDeg**(δ, μ, i, dim) and **DecreaseDeg**(δ, μ, i, dim) requires $O(dim)$ time.

IncreaseDeg(δ, μ, i, dim)
```
IF (i ≠ dim AND δ[i] + 1 = δ[i + 1]) THEN
    μ[i] ← μ[i] − 1;
    μ[i + 1] ← μ[i + 1] + 1;
    IF (i ≠ 0 AND μ[i] = 0) THEN
        FOR k = i TO dim − 1 DO
            δ[k] ← δ[k + 1];
            μ[k] ← μ[k + 1];
        dim ← dim − 1
ELSE
    IF (i ≠ 0 AND μ[i] = 1) THEN δ[i] ← δ[i] + 1;
    ELSE
        μ[i] ← μ[i] − 1;
        FOR k = dim DOWNTO i + 1 DO
            δ[k + 1] ← δ[k];
            μ[k + 1] ← μ[k];
        δ[i + 1] ← δ[i] + 1;
        μ[i + 1] ← 1;
        dim ← dim + 1
RETURN(δ, μ, dim).
```

DecreaseDeg(δ, μ, i, dim)
```
IF (δ[i] − 1 = δ[i − 1]) THEN
    μ[i] ← μ[i] − 1;
    μ[i − 1] ← μ[i − 1] + 1;
    IF (μ[i] = 0) THEN
        FOR k = i + 1 TO dim DO
            δ[k − 1] ← δ[k];
            μ[k − 1] ← μ[k];
        dim ← dim − 1
ELSE
    IF (μ[i] = 1) THEN δ[i] ← δ[i] − 1;
    ELSE
        μ[i] ← μ[i] − 1;
        FOR k = dim DOWNTO i DO
            δ[k + 1] ← δ[k];
            μ[k + 1] ← μ[k];
        δ[i] ← δ[i + 1] − 1;
        μ[i] ← 1;
        dim ← dim + 1
RETURN(δ, μ, dim).
```

Let us now consider a threshold graph G. Let (x, y), $x \in D_i$ and $y \in D_j$, the edge to add/delete to G. Items 2 and 3 of Lemma 6 give a characterization of the indices i and j to ensure that the modified graph is still a threshold graph; namely, $i + j = m$ in case of insertion, and $i + j = m + 1$ in case of deletion.

We present two operations, **InsertEdge**(δ, μ, i, m) and **DeleteEdge**(δ, μ, i, m), that update the data structure when an edge is added between a node in box D_i and a node in box D_{m-i} and when an edge is deleted between a node in box D_i and a node in box D_{m+1-i}, respectively.

Observe that with the insertion of an edge, the degrees of its endpoints are increased by one. Thus we can call twice subroutine **IncreaseDeg**, once on a node in box D_i and once on a node in box D_{m-i}. We have just to take into account that the increment of the degree of node in box D_i can change the index of the box of the other endpoint. Analogous considerations hold for the deletion of an edge. These observations give rise to the following simple algorithms:

InsertEdge(δ, μ, i, m)
  $j \leftarrow m - i$;
  $a \leftarrow m$;
  IncreaseDeg(δ, μ, i, m);
  $CASE(m - a)$
    -1 : **IncreaseDeg**$(\delta, \mu, j - 1, m)$;
    0 : **IncreaseDeg**(δ, μ, j, m);
    $+1$: **IncreaseDeg**$(\delta, \mu, j + 1, m)$;

DeleteEdge(δ, μ, i, m)
  $j \leftarrow m + 1 - i$;
  $a \leftarrow m$;
  DecreaseDeg(δ, μ, i, m);
  $CASE(m - a)$
    -1 : **DecreaseDeg**$(\delta, \mu, j - 1, m)$;
    0 : **DecreaseDeg**(δ, μ, j, m);
    $+1$: **DecreaseDeg**$(\delta, \mu, j + 1, m)$;

Since after the execution of **IncreaseDeg** (**DecreaseDeg**), m may potentially vary from m to $m \pm 1$, with the execution of **InsertEdge** (**DeleteEdge**) the number of boxes can potentially vary from m to $m \pm 2$. In Figure 2 we show that all the five possibilities may actually occur.

Assume now that G is a difference graph.

The algorithms for adding/eliminating an edge in G are based on the same idea presented for the algorithms on threshold graphs, but they are even simpler because the data structure used for representing these graphs keeps separated the bipartition (and so, adding a new box after the first call of **IncreaseDeg** does not affect the index of the other endpoint).
Notice that Item 1 of Lemma 7 ensures that the number of boxes in the two classes is the same t. So, for difference graphs, t can either remain unaltered or to change to $t \pm 1$ and all the three possibilities may actually occur.

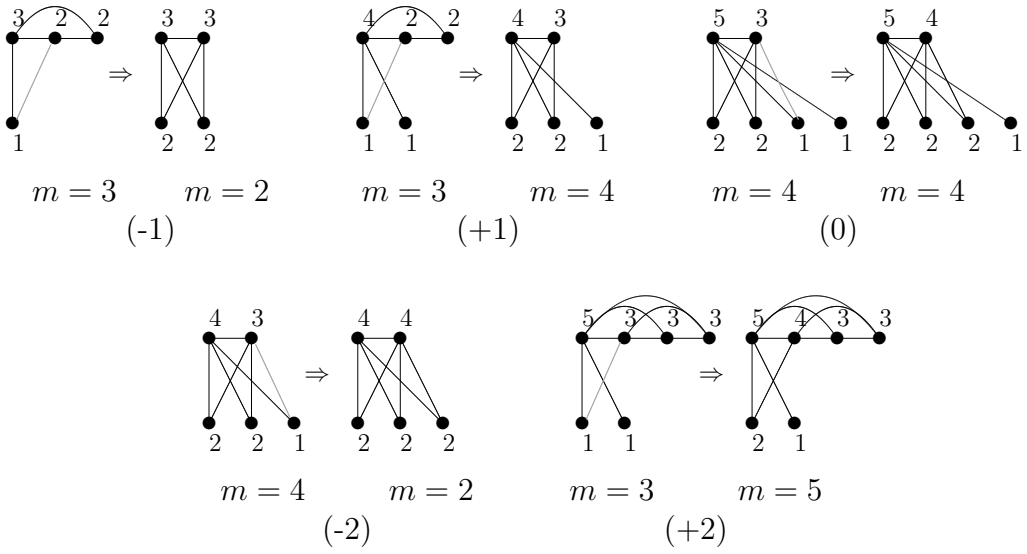

Figure 2: Examples proving that all 5 cases in algorithm **InsertEdge** are possible. Grey edges represent the edges that are going to be added. (In order to consider **DeleteEdge**, figures must be read from right to left.)

The two algorithms for inserting and deleting an edge in a difference graph follow:

D − **InsertEdge**$(\delta_U, \mu_U, \delta_W, \mu_W, i, t)$
  $j \leftarrow t - i$;
  IncreaseDeg(δ_U, μ_U, i, t);
  IncreaseDeg$(\delta_W, \mu_W, j, t,)$;

D − **DeleteEdge**$(\delta_U, \mu_U, \delta_W, \mu_W, i, t)$
  $j \leftarrow t + 1 - i$;
  DecreaseDeg(δ_U, μ_U, i, t);
  DecreaseDeg(δ_W, μ_W, j, t);

Notice that, in view of all the reasonings done, all the algorithms described in this section are correct and maintain the minimality of the integral separators. Moreover, they require time linear w.r.t. the number of different degrees in the graph.

5. Adding/deleting a node to threshold/difference graphs

In this section we will work with nodes in an analogous way as we did in Section 4 with edges. Also in this case, we keep immediately available the knowledge of the minimum separator for the new graph. We start defining four functions, operating on the data structures introduced in Section 3.

By $+\mathbf{Node}(\delta, \mu, d, dim)$ we denote the operation of giving space to a new node of degree d either in a threshold graph or in a partition of a difference graph, without caring about the update of its neighbors (that will be done with another subroutine). This subroutine looks for the box where the new node must be inserted: if there exists a box D_i with degree d, μ_i is simply increased by one; otherwise a new box for the new node is created.

By $\mathbf{IncreaseDegreeOfSetNode}(\delta, \mu, d, dim)$ we denote the operation of augmenting by one the degree of the d nodes of highest degree either in a threshold graph or in a partition of a difference graph. This subroutine increases by one the degree of all the boxes D_i s.t. $d - \sum_{s=j+1}^{m} |D_s| \geq 0$, while nodes of boxes $D_1, \ldots D_{j-1}$ remain unchanged. For what concerns D_j, it is in general split into two boxes (precisely $d - \sum_{s=j+1}^{m} |D_s|$ nodes leave D_j to form a new box with degree augmented by one).

We can define even the symmetric functions: by $\mathbf{-Node}(\delta, \mu, i, dim)$ we denote the operation eliminating from the data structure storing either a threshold or a difference graph a node in box D_i, $0 \leq i \leq dim$, regardless of its neighbors (whose degree will be updated with another subroutine).

By $\mathbf{DecreaseDegreeOfSetNode}(\delta, \mu, d, dim)$ we denote the operation of decreasing by one the degree of the d nodes of highest degree either in a threshold graph or in a partition of a difference graph.

The execution of all these subroutines requires $O(dim)$ time.

```
+Node(δ, μ, d, dim))
  i ← 0;
   WHILE(i ≤ dim AND d > δ[i]) DO i = i + 1;
  IF (i ≤ dim AND d = δ[i]) THEN μ[i] ← μ[i] + 1;
      ELSE
            dim ← dim + 1;
            FOR k = dim DOWNTO i + 1 DO
                δ[k] ← δ[k − 1];
                μ[k] ← μ[k − 1];
            δ[i] ← d;
            μ[i] ← 1;
  RETURN(δ, μ, dim).
```

```
−Node(δ, μ, i, dim)
  μ[i] ← μ[i] − 1;
  IF (i ≠ 0 AND μ[i] ≠ 0) THEN
                FOR k = i TO dim − 1 DO
                    δ[k] ← δ[k + 1];
                    μ[k] ← μ[k + 1];
                dim ← dim − 1;
  RETURN(δ, μ, dim).
```

IncreaseDegreeOfSetNode(δ, μ, d, dim)
$i \leftarrow dim; a \leftarrow d;$
WHILE $(a \geq \mu[i])$ DO
$a \leftarrow a - \mu[i];$
$\delta[i] \leftarrow \delta[i] + 1;$
$i = i - 1;$
IF $(a \neq 0)$ THEN
$dim \leftarrow dim + 1;$
FOR $j = dim$ DOWNTO $i + 2;$
$\mu[j] \leftarrow \mu[j - 1];$
$\delta[j] \leftarrow \delta[j - 1];$
$\delta[i + 1] \leftarrow \delta[i] + 1;$
$\mu[i + 1] \leftarrow a;$
$\mu[i] \leftarrow \mu[i] - a;$
$RETURN(\delta, \mu, dim)$.

DecreaseDegreeOfSetNode(δ, μ, d, dim)
$i \leftarrow dim; a \leftarrow d;$
WHILE $(a > 0)$ DO
$a \leftarrow a - \mu[i];$
$\delta[i] \leftarrow \delta[i] - 1;$
$i \leftarrow i - 1;$
IF $(i \neq dim$ AND $\delta[i] = \delta[i + 1])$ THEN
$\mu[i] \leftarrow \mu[i] + \mu[i + 1]$
FOR $k = i + 1$ TO $dim - 1$ DO
$\delta[k] \leftarrow \delta[k + 1];$
$\mu[k] \leftarrow \mu[k + 1];$
$dim \leftarrow dim - 1;$
$RETURN(\delta, \mu, dim)$.

Let now $G = (V, E)$ be a threshold graph. Adding a new node of degree d to G yields a threshold graph if and only if the d neighbors of the new node are the d nodes with highest degrees (this can be easily deduced from Item 4 of Lemma 6). So, we can call **IncreaseDegreeOfSetNode** and observe that, after its execution, m could be increased by one. Then, we have to update the data structure by inserting the new node by means of **+Node**, and even in this case m could be increased by one. So, the number of the different degrees can potentially vary from m to $m + 2$. Figure 3 shows that all three possibilities can occur.

By **InsertNode**(δ, μ, d, m) we denote the operation of updating the data structure storing threshold graph G when a node of degree d is added to the graph.

The previous reasonings can be repeated when G is a difference graph (assuming, w.l.o.g., that the new node is inserted in partition U), so giving rise to **D-InsertNode**$(\delta_U, \mu_U, \delta_W, \mu_W, i, t)$, that is the operation of updating the data structure when a node of degree d is added to the difference graph.

InsertNode(δ, μ, d, m)
+Node(δ, μ, d, m)
IncreaseDegreeOfSetNode(δ, μ, d, m)

D − InsertNode$(\delta_U, \mu_U, \delta_W, \mu_W, d, t)$
+Node(δ_U, μ_U, d, t)
IncreaseDegreeOfSetNode(δ_W, μ_W, d, t)

Now we consider the problem of deleting nodes to a threshold graph.

Any node-induced subgraph G' of G is a threshold graph (indeed for the graph G' use the mapping a restricted on the nodes of G' and the same value S). Thus the class of threshold graphs is closed under the deletion of an arbitrary node.

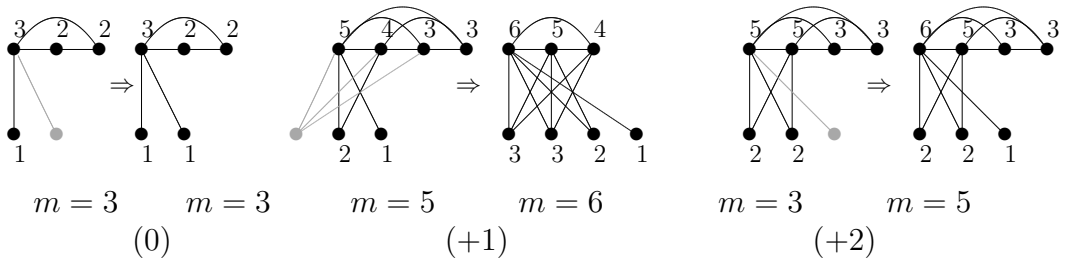

Figure 3: Examples proving that all 3 cases in algorithm **InsertNode** are possible. Grey nodes and edges represent the objects that are going to be added (in order to consider **DeleteNode** figure must be read from right to left).

Given a threshold graph, by **DeleteNode**(δ, μ, i, m) we denote the operation of updating the data structure when a node is deleted from box D_i, $0 \leq i \leq m$. This deletion is performed by **-Node** that can have as consequence the disappearance of box D_i (if $i \neq 0$ and $|D_i| = 1$). Thus m can decrease by one. Moreover, the $\delta[i]$ nodes with highest degree must have their degrees decreased by one. These nodes belong to boxes $D_m, \ldots, D_{m+1-i}$. It can occur that the degree of nodes in box $m+1-i$ becomes equal to the degree of the nodes in box $m-i$ and, in this case, the two boxes merge and the number of boxes further decrease by one. Hence after deleting a node, the number of boxes in the degree partition can potentially vary from m to $m-2$ and all cases can occur, as shown in Figure 3.

Analogous reasonings can be done when G is a difference graph, and define **D-DeleteNode**$(\delta_U, \mu_U, \delta_W, \mu_W, i, t)$ as the operation of updating the data structure when a node is deleted (assuming w.l.o.g. that the new node is deleted from partition U).

DeleteNode(δ, μ, i, m)
  $d \leftarrow \delta[i]$
  −Node(δ, μ, i, m)
  DecreaseDegreeOfSetNode(δ, μ, d, m)

D − deleteNode$(\delta_U, \mu_U, \delta_W, \mu_W, i, t)$
  $d \leftarrow \delta_U[i]$
  −Node(δ_U, μ_U, i, t)
  DecreaseDegreeOfSetNode(δ_W, μ_W, d, t)

Also in this case, all the algorithms described in this section are correct and maintain the minimality of the integral separators. Moreover, they require time linear w.r.t. the number of different degrees in the graph.

6. A data structure for threshold signed graphs

In this section we specify the data structure used for representing threshold signed graphs.

Given a threshold signed graph G, let us call G^- and G^+ the threshold graphs induced by node sets $X = \{x \in V \text{ s.t. } a(x) < 0\}$ and $Y = \{x \in V \text{ s.t. } a(x) \geq 0\}$, respectively, and let D be the difference graph constituted by all the edges of G between X and Y. We consider the degree of each node v in G, $\deg(v)$, as the sum of the degree in the threshold subgraph to which it belongs to, $\deg_{|G^-}(v)$ or $\deg_{|G^+}(v)$, plus the degree in the difference subgraph $\deg_{|D}(v)$.

Given a graph G, we recall that the *neighborhood* of a node v is the set of all neighbors of v, and its *closed neighborhood* is the neighborhood of v plus node v itself. Two nodes u and w are *false twins* if they have the same neighborhood; they are *true twins* if they have the same closed neighborhood. We say that u and w are simply *twins* if they are either true or false twins and they belong to the same set, X or Y.

Even though there is not a tie between the degree partition of a threshold signed graph and its structure, as in the case of threshold and difference graphs, it is possible to extend the reasonings done in the proof of Theorem 9 to this class of graphs. Indeed, if v is an isolated node it is not restrictive to assume $a(v) = 0$ and, obviously, if $a(v) = 0$ then v is an isolated node. Moreover, it is easy to see that two nodes having the same value of a are necessarily twins. From the other hand, if there are two twins u and w having $a(u) \neq a(w)$ (w.l.o.g. let $a(u) < a(w)$), we can easily modify function a in order to assign them the same value (that is $a(w)$ if u and w are connected and $a(u)$ otherwise).

So, from now on, we consider only node weight functions assigning value 0 to each isolated node and the same value to each set of twins.

Let us consider the partition of the node set into equivalence classes, $B_1, \ldots, B_\Delta$, induced by the relation of being twins.

Before proving that Δ is linear w.r.t. the sum of the number of different degrees in X and Y, we need to recall two important properties.

Lemma 12. *[6] Given a threshold signed graph, for each pair of nodes $u, v \in X$ (respectively $u, v \in Y$) it holds:*

$deg_{|G^-}(u)$ (respectively $deg_{|G^+}(u)$) $\geq$ $deg_{|G^-}(v)$(respectively $deg_{|G^+}(v)$) if and only if $deg_{|D}(u) \geq deg_{|D}(v)$.

Lemma 13. *[16] Given a graph $G = (V, E)$, G is a threshold graph if and only if the neighborhoods of its nodes form a unique chain of inclusions; G is a differene graph if and only if the neighborhoods of its nodes form two chains of inclusions (inducing X and Y).*

Theorem 14. *Given a threshold signed graph G and the partition into equivalence classes, $B_1, \ldots, B_\Delta$, induced by the relation of being twins, Δ is linear w.r.t. the sum of the number of different degrees in X and Y.*

PROOF. Let u and v be two nodes of the threshold signed graph s.t. $\deg(u) = \deg(v)$, i.e. $\deg_{|G^-}(u)+\deg_{|D}(u) = \deg_{|G^-}(v)+\deg_{|D}(v)$. The previous lemma implies $\deg_{|G^-}(u) = \deg_{|G^-}(v)$ and $\deg_{|D}(u) = \deg_{|D}(v)$. If u and v are both in X (respectively, Y), from Lemma 13, it follows that u and v are either true or false twins, and hence are twins. Conversely, if u is in X and v is in Y, u and v cannot be twins, even if they have the same degree.

Hence, Δ is bounded by the number of different degrees of the nodes in X plus the the number of different degrees of the nodes in Y. □

As consequence of all these reasonings, we may store a threshold signed graphs by means of two arrays $\alpha[0..\Delta]$ and $\mu[0..\Delta]$: in $\alpha[i]$ there is the value of the weight assigned to the $\mu[i]$ nodes of B_i, $0 \leq i \leq \Delta$; if there are no isolated nodes $\alpha[0]$ and $\mu[0]$ are set to 0. Variables S and T store the two thresholds.

7. Disjoint union and join of two threshold graphs

Given two graphs with disjoint node sets $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, their *disjoint union* is the graph $G_1 \cup G_2 = (V_1 \cup V_2, E_1 \cup E_2)$; their *join* is the graph $G_1 + G_2$ obtained adding to their disjoint union all the edges that connect the nodes of the first graph with the nodes of the second graph.

We observe that, if G_1 and G_2 are both threshold graphs, $G_1 \cup G_2$ and $G_1 + G_2$ are threshold signed graphs, where the difference graph connecting the two threshold graphs is either the null graph (in $G_1 \cup G_2$) or the complete bipartite graph (in $G_1 + G_2$). In this section we restrict our attention to the operations of disjoint union and join between threshold graphs since the result of the disjoint union and the join of two difference graphs do not fall in general in any of the classes considered in this paper.

Let us now show how to generate the data structures representing the threshold signed graphs, $G_1 \cup G_2$ and $G_1 + G_2$.

Let $\delta_1[0..m_1]$, $\mu_1[0..m_1]$ and $\delta_2[0..m_2]$, $\mu_2[0..m_2]$ be the arrays storing G_1 and G_2, respectively. Assume first that G_1 and G_2 have the same threshold S (i.e. $m_1 = m_2 = m$).

Informally, the array $\alpha[1..\Delta]$ of both $G_1 \cup G_2$ and $G_1 + G_2$ is obtained by opportunely transcribing the values of the node weight function of the single threshold graphs (deduced through Theorem 9), the array $\mu[1..\Delta]$ is obtained by copying the values of μ_1 and μ_2, while threshold S is kept unaltered. For what concerns threshold T, in the case of $G_1 \cup G_2$ it is set to a sufficiently large value in order to guarantee that no edges are in the difference subgraph, in the case of $G_1 + G_2$ it is set to a sufficiently small value in order to guarantee that the difference subgraph is a complete bipartite graph. In this latter case, T assumes a too small value, contradicting Property 5 of Definition 8, so we need to modify the values of the node weight function and of the two thresholds in order to restore the property.

The following lemmas formalize the operations of disjoint union and join of threshold graphs, and guarantee their correctness.

Lemma 15. *Let be given two threshold graphs G_1 and G_2 by means of $\delta_1[0..m]$, $\mu_1[0..m]$ and $\delta_2[0..m]$, $\mu_2[0..m]$ and let $S = m+1$ be their common threshold. It is possible to determine the threshold signed graph $G_1 \cup G_2$ in time linear w.r.t. the number of its different degrees.*

Proof. Consider the following function:

DisjointUnion$(\delta_1, \mu_1, \delta_2, \mu_2, m)$
  $\mu[0] \leftarrow \mu_1[0] + \mu_2[0]$; $\alpha[0] \leftarrow 0$;
  FOR $i = 1$ TO m DO
    $\alpha[i] \leftarrow -i$; $\mu[i] \leftarrow \mu_1[i]$;
  FOR $i = 1$ TO m DO
    $\alpha[m+i] \leftarrow i$; $\mu[m+i] \leftarrow \mu_2[i]$;
  $\Delta \leftarrow 2m$;
  $S \leftarrow m+1$;
  $T \leftarrow 2m+1$;
RETURN $(\alpha, \mu, S, T, \Delta)$.

Both S and T are greater than the modulo of each $\alpha[i]$, $i = 0, \ldots, \Delta$ as far as they are defined.

Moreover, the two threshold subgraphs G^- and G^+ of $G_1 \cup G_2$ are exactly the same as G_1 and G_2, respectively. Finally, no edge can satisfy the condition $\alpha[u] + \alpha[v] \geq T$ in view of the definition of T, so the difference subgraph D is empty. It follows that α, μ, S and T correctly define $G_1 \cup G_2$.

We conclude by observing that function **DisjointUnion** runs in $O(\Delta)$ time that is linear w.r.t. the number of different degrees in $G_1 \cup G_2$ for Theorem 14. □

Lemma 16. *Let be given two threshold graphs G_1 and G_2 by means of $\delta_1[0..m]$, $\mu_1[0..m]$ and $\delta_2[0..m]$, $\mu_2[0..m]$ and let $S = m+1$ be their common threshold. It is possible to determine the threshold signed graph $G_1 + G_2$ in time linear w.r.t. the number of its different degrees.*

Proof. Consider the following function:

Join$(\delta_1, \mu_1, \delta_2, \mu_2, m)$
  $\alpha[0] \leftarrow 0$; $\mu[0] \leftarrow 0$;
  IF $\mu_1[0] \neq 0$
    THEN flag1 $\leftarrow 0$
    ELSE flag1 $\leftarrow 1$
  FOR $i =$flag1 TO m DO
    $\alpha[i + 1 -$ flag1$] \leftarrow -i$; $\mu[i + 1 -$ flag1$] \leftarrow \mu_1[i]$;
  IF $\mu_2[0] \neq 0$
    THEN flag2 $\leftarrow 0$
    ELSE flag2$\leftarrow 1$
  FOR $i =$flag2 TO m DO
    $\alpha[m + i + 2 -$ flag1-flag2$] \leftarrow i$; $\mu[m + i + 2 -$ flag1-flag2$] \leftarrow \mu_2[i]$;
  $\Delta \leftarrow 2m + 2-$flag1-flag2;
  $S \leftarrow m + 1$;
  $T \leftarrow \min_{1 \leq i \leq \Delta}\{|\alpha[i]|\}$;
  $k \leftarrow m - T + 1$;
  FOR $i = 1$ TO $m + 1 - flag1$ DO
    $\alpha[i] \leftarrow \alpha[i] - k$;
  FOR $i = m + 2-$flag1 TO $2m + 2-$flag1-flag2 DO
    $\alpha[i] \leftarrow \alpha[i] + k$;
  $S \leftarrow S + 2k$;
  $T \leftarrow T + 2k$;
RETURN $(\alpha, \mu, S, T, \Delta)$.

Preliminarily, observe that $G_1 + G_2$ cannot have isolated nodes, so we set $\mu[0]$ to 0; moreover, if either G_1 or G_2 contain isolated nodes, a box needs to be added: we do this exploiting the two boolean variables flag1 and flag2.

So, even in this case, the two threshold graphs G^- and G^+ of $G_1 + G_2$ are exactly the same as G_1 and G_2, respectively. T is set to the modulo of the smallest node weight in order to guarantee that D is a complete bipartite graph. In this way, T results in a value that contradicts Property 5 of Definition 8. By incrementing the modulo of each $\alpha[i]$ of an opportune value k and S and T by $2k$, we are able to restore the inequality.

It is immediate to see that function **Join** runs in $O(\Delta)$ time that is linear w.r.t. the number of different degrees in $G_1 + G_2$ for Theorem 14. □

It remains to handle the case in which G_1 and G_2 have different thresholds. In this case, we prepose to the functions described in the proofs of Lemmas 15 and 16 a preprocessing phase that equalize their thresholds, as detailed in the following lemma, where with the notation $a' = xa + y$ (where x and y are integer values and a is a node weight function) we compactly mean that, for each node v, $a'(v) = xa(v) + y$. We want to underline that now on we represent a threshold graph in terms of its separator, instead of in terms of our data structure, because the description of the equalization appears more comprehensive.

Lemma 17. *Let be given two thresholds graphs G_1 and G_2 and let (a_1, S_1) and (a_2, S_2) be their integral separators with $S_1 < S_2$. Then $(a'_1, S'_1) = (2a_1 + S_2 - S_1, 2S_2)$ is a integral separator for G_1 and $(a'_1, a'_2) = (2a_2, 2S_2)$ is an integral separator for G_2.*

Proof. Let (v, w) be an edge in G_1, i.e. $a_1(v) + a_1(w) \geq S_1$; then $a'_1(v) + a'_1(w) = 2a_1(v) + S_2 - S_1 + 2a_1(w) + S_2 - S_1 \geq 2S_2 = S'_1$. In the same way, let v and w be not connected in G_1, i.e. $a_1(v) + a_1(w) < S_1$; then $a'_1(v)+a'_1(w) = 2a_1+S_2-S_1 < 2S_2 = S'_1$. Finally, the pair (a'_1, S'_1) is a feasible integral separator since, for any node v, $a_1(v) < S_1$ implies $a'_1(v) < S'_1$.

Analogous reasonings lead to prove that (a'_2, S'_2) is an integral separator for G_2. □

From Lemmas 15, 16 and 17, observing that the operations required by Lemma 17 in order to equalize thresholds S_1 and S_2 must be performed directly on array α in order to keep the time complexity bounded by $O(\Delta)$, we can conclude underlying that all the algorithms described in this section are correct and require time linear w.r.t. the number of different degrees in the graph. Nevertheless, the values of the node weight function and of the two thresholds S and T of the resulting threshold signed graphs will be integral but not necessarily minimal.

8. Complement of threshold signed graphs

Given a graph $G = (V, E)$, its *complement* $\bar{G}$ has the same node set V of G and two nodes are adjacent if and only if they are not adjacent in G.

The classes of threshold and threshold signed graphs are closed under complement [3] while the complement of a difference graph is a threshold signed graph. So, it is worth to be solved the problem of computing the weight function and the thresholds of the complement of one of these graphs without recomputing them from scratch by using, for example, a recognizing algorithm.

Observe that, given a graph G that is either a threshold or a difference or a threshold signed graph, all its nodes that have the same degree d in G have the same degree $|V| - 1 - d$ in $\bar{G}$. If we exploit this observation to compute the complement of a threshold graph G stored through the data structure described in Section 3, nodes with the same degree are in in the same box, and so it is easy to efficiently construct arrays $\bar{\delta}$ and $\bar{\mu}$ in $O(m)$ time (see function **ThresholdComplement**).

ThresholdComplement(δ, μ, m)
  $n = 0$;
  FOR $i = 0$ TO m DO
    $n = n + \mu[i]$;
  $i = 0; j = m$;
  IF $\mu[0] = 0$ THEN
    FOR $i = 0$ TO m DO
      $\bar{\delta}[m - i] \leftarrow n - 1 - \delta[i]$;
      $\bar{\mu}[m - i] \leftarrow \mu[i]$;
    $\bar{m} \leftarrow m - 1$;
  ELSE
    $\delta[0] \leftarrow 0$; $\bar{\mu}[0] \leftarrow 0$;
    FOR $i = 0$ TO m DO
      $\bar{\delta}[m - i + 1] \leftarrow n - 1 - \delta[i]$;
      $\bar{\mu}[m - i + 1] \leftarrow \mu[i]$;
    $\bar{m} \leftarrow m + 1$;
  RETURN $(\bar{\delta}, \bar{\mu}, \bar{m})$.

Vice-versa, the data structure for threshold signed graphs described in Section 6 does not store the degree of the nodes, but function a; in order to compute $\bar{a}$, $\bar{S}$ and $\bar{T}$ of $\bar{G}$ starting from the degrees of the nodes of $\bar{G}$, we should run the fastest known algorithm [3], working in $O(n^2)$ time, where n is the number of nodes of G.

In this section, we provide a method to determine and store the complement of a threshold signed graph G in time linear w.r.t. the number of its different degrees.

Theorem 18. *Let be given a threshold signed graph $G = (X \cup Y, a, S, T)$ stored through arrays $a[0..\Delta]$ and $\mu[0..\Delta]$ and the two values S and T. It is possible to determine the threshold signed graph $\bar{G} = (X \cup Y, \bar{a}, \bar{S}, \bar{T})$ in time linear w.r.t. the number of different degrees ($O(\Delta)$).*

Proof. First observe that if G is a complete graph, then it is easy to build the data structure storing $\bar{G}$ in constant time. So, it is not restrictive to assume that at least one edge is missing from G. Consider the following function:

Complement(a, μ, δ, S, T)
  $P^* \leftarrow \max\{S, T\} - \min_{1 \le i \le n}\{|a_i|\}$;
  $\epsilon \leftarrow \min_{1 \le i,j \le n, (v_i,v_j) \notin E}\{S - |a_i + a_j|, T - |a_i - a_j|\}$;
  FOR $i = 0$ TO Δ DO
    IF $a[i] \ge 0$ THEN $\bar{a}[i] \leftarrow -(P^* - a[i])$
    ELSE $\bar{a}[i] \leftarrow (P^* - |a[i]|)$
    $\bar{\mu}[i] \leftarrow \mu[i]$
  $\bar{T} \leftarrow 2P^* - T + \epsilon$
  $\bar{S} \leftarrow 2P^* - S + \epsilon$
RETURN$(\bar{a}, \bar{\mu}, \Delta, \bar{S}, \bar{T})$.

Notice that the value of ϵ is well defined in view of the hypothesis that G is not a complete graph.

We have to prove that the assigned weights and the thresholds are feasible values and that they define exactly the complement of the original graph.

Let us prove first that $|\bar{a[i]}| < \min\{\bar{S}, \bar{T}\}$ for each $i = 1, \ldots, n$. From the definitions of $\bar{S}$ and $\bar{T}$ we have that both of them are $\ge 2P^* - \max\{S, T\} + \epsilon$ hence, in particular:

$$\min\{\bar{S}, \bar{T}\} \ge 2P^* - \max\{S, T\} + \epsilon > 2P^* - \max\{S, T\}. \tag{7}$$

From the other hand, $P^* = \max\{S, T\} - \min_{1 \le i \le n}\{|a_i|\} \ge \max\{S, T\} - |a[i]|$. From this inequality and from (7) we get that $\min\{\bar{S}, \bar{T}\} > P^* - |a[i]| = |\bar{a[i]}|$.

It remains to prove that $(v_i, v_j) \in \bar{E}$ if and only if either $|\bar{a[i]} + \bar{a[j]}| \ge S$ or $|\bar{a[i]} - \bar{a[j]}| \ge T$. We divide the proof into two sub-cases, according to whether (v_i, v_j) belongs to $\bar{E}$ or not.

Assume first that $(v_i, v_j) \in \bar{E}$. This means that $(v_i, v_j) \notin E$, i.e. $|a[i] + a[j]| < S$ and $|a[i] - a[j]| < T$. As far as ϵ has been defined, it also holds that:

$$|a[i] + a[j]| \leq S - \epsilon \text{ and } |a[i] - a[j]| \leq T - \epsilon. \tag{8}$$

If $a[i]$ and $a[j]$ have the same sign, then $|\bar{a[i]} + \bar{a[j]}| = 2P^* - |a[i] + a[j]| \geq 2P^* - S + \epsilon = \bar{S}$ in view of (8) and of the definition of $\bar{S}$.

If $a[i]$ and $a[j]$ have opposite sign, then $|\bar{a[i]} - \bar{a[j]}| = 2P^* - |a[i] - a[j]| \geq 2P^* - T + \epsilon = \bar{T}$ in view of (8) and of the definition of $\bar{T}$.

Assume now that $(v_i, v_j) \notin \bar{E}$. This means that $(v_i, v_j) \in E$, i.e. $|a[i] + a[j]| \geq S$ or $|a[i] - a[j]| \geq T$.

If $|a[i] + a[j]| \geq S$ it means that a_i and a_j have the same sign, and hence $\bar{a[i]}$ and $\bar{a[j]}$ have the same sign, too. So $|\bar{a[i]} + \bar{a[j]}| = 2P^* - |a[i] + a[j]| \leq 2P^* - S < 2P^* - S + \epsilon = \bar{S}$, so proving that (v_i, v_j) is not an edge of $\bar{G}$.

If, finally, $|a[i] - a[j]| \geq T$ it means that $a[i]$ and $a[j]$ have different sign, and hence $\bar{a[i]}$ and $\bar{a[j]}$ have different sign, too. So $|\bar{a[i]} - \bar{a[j]}| = 2P^* - |a[i] - a[j]| \leq 2P^* - T < 2P^* - T + \epsilon = \bar{T}$, so proving that also in this case (v_i, v_j) is not an edge of $\bar{G}$.

It is immediate to see that function **Complement** runs in $O(\Delta)$ time, that is linear w.r.t. the number of different degrees in $\bar{G}$ for Theorem 14. □

We conclude this section observing that the integral separator we deduce for $\bar{G}$ is minimum if G is a threshold graph, while it is in general not minimum if G is a threshold signed graph.

9. Conclusions

In this paper we have studied how to dynamically operate on threshold, difference and threshold sign graphs and we provided efficient solutions, working in time linear w.r.t. the number of the different degrees of the input graph. In order to do this we proposed simple data structures to store these graphs and studied the problems of adding/deleting edges/nodes, the disjoint union, the join and the complement. In the case of addition/deletion of edges/nodes we provided efficient solutions only for the classes of threshold and difference graphs; indeed, it does not seem possible to us to modify function a and thresholds S and T of a threshold signed graph to which an

edge/node is added/deleted without recomputing it from scratch (that would take $O(n^2)$ time). Moreover, we compute a minimum integral separator for threshold and difference graphs in time that is linear w.r.t. the number of different degrees, so improving some previous results [20, 21], minimizing other weight functions raised by other (equivalent) definitions and running in linear time w.r.t. the number of nodes.

References

[1] M. Andelić and S.K. Simić, Some notes on the threshold graphs. *Discrete Math.* 310(17–18), 2241–2248, 2010.

[2] Z. Bao-Lin. Difference Graphs of Block ADI Method, *SIAM J. on Numerical Analysis* 38(3), 742–752, 2000.

[3] C. Benzaken, P.L. Hammer and D. de Werra, Threshold characterization of graphs with Dilworth number two. *J. Graph Theory* 9, 245–267, 1985.

[4] A. Bhattacharya, U. N. Peled, and M. K. Srinivasan, Cones of closed alternating walks and trails. *Linear Algebra and its Applications* 423, 351–365, 2007.

[5] A. Bhattacharya, S. Sivasubramaniam, and M. K. Srinivasan, The polytope of degree partitions. *Electronic Journal of Combinatorics*13: #R46, 2006.

[6] T. Calamoneri and R. Petreschi. On Pairwise Compatibility Graphs having Dilworth Number k. *Theoretical Computer Science* 547, 82–89, 2014.

[7] V. Chvátal and P.L. Hammer. Aggregation of inequalities in integer programming, Proc. Worksh. Bonn 1975, *Annals of Discrete Mathematics* 1, Amsterdam: North-Holland, 145–162, 1975 .

[8] O. Cogis. Ferrers digraphs and threshold graphs. *Discrete Mathematics* 38, 33–46, 1982.

[9] A. Delugach, H. de Moor: Difference graphs. *Proc. ICCS 2005*, 41–53, 2005.

[10] J.-P. Doignon, A. Ducamp, and J.-C. Falmagne. On realizable biorders and the biorder dimension of a relation. *Journal of Mathematical Psychology* 28, 73–109, 1984.

[11] K. Ecker and S. Zaks. On a graph labelling problem. Bericht 99, Gesellschraft für Mathematik und Datenverarbeitung MBH, Bonn 1977.

[12] P. Heggernes and C. Papadopoulos. Single-edge monitor sequences of graphs and linear-time algorithms for minimal completions and deletions. *Theoretical Computer Science* 410, 1–15, 2009.

[13] P.B. Henderson and Y. Zalcstein. A graph-theoretic characterization of the PV_{chunk} class of synchronizing primitives. *SIAM Journal of Computing* 6, 88–108, 1977.

[14] G. J. Koop. Cyclic scheduling of offweekends. *Operations Research Letters* 4, 259–263, 1986.

[15] M. Koren. Extreme degree sequences of simple graphs. *Journal of Combinatorial Theory* 2, 253–276, 1972.

[16] N.V.R. Mahadev and U.N. Peled. Threshold Graphs and Related Topics, *Annals of Discrete Mathematics* 56, North-Holland, Amsterdam, 1995.

[17] K. Makino, Y. Uno and T. Ibaraki. Minimum Edge Ranking Spanning Trees of Threshold Graphs, *Proc. ISAAC 2002*, LNCS 2518, 428–440, 2002.

[18] V. Neiger, C. Crespelle, E. Fleury. On the Structure of Changes in Dynamic Contact Networks. Proc. SITIS '12 731–738, 2012.

[19] I. Nor, J. Engelstädter, O. Duron, M. Reuter, M.-F. Sagot and S. Charlat. On the Genetic Architecture of Cytoplasmic Incompatibility: Inference from Phenotypic Data, *The American Naturalist* 182(1), 15–24, 2013.

[20] E.T. Ordman. Threshold coverings and resource allocation. *16th Southeastern Conference on Combinatorics, Graph Theory and Computing*, 99–113, 1986.

[21] J. Orlin. The minimal integral separator of a threshold graph. *Annals of Discrete Mathematics* 1, Amsterdam: North-Holland, pp. 415–419, 1977.

[22] R. Shamir and R. Sharan. A fully dynamic algorithm for modular decomposition and recognition of cographs. *Discrete applied Mathematics* 136, 329–340, 2004.

[23] M. Yannakakis. The complexity of the partial order dimension problem. *SIAM Journal on algebraic and Discrete Methods* 3, 351–358, 1982.