

# A multivariate interlace polynomial

Bruno Courcelle  
Bordeaux University and CNRS, LaBRI

December 2006 (revised March 2007)

## Abstract

We define a multivariate polynomial that generalizes several *interlace* polynomials defined by Arratia, Bollobas and Sorkin on the one hand, and Aigner and van der Holst on the other. We follow the route traced by Sokal, who defined a multivariate generalization of Tutte's polynomial. We also show that bounded portions of our interlace polynomial can be evaluated in polynomial time for graphs of bounded clique-width. Its full evaluation is necessarily exponential just because of the size of the result.

**Keywords** : Interlace polynomial, multivariate polynomial, monadic second-order logic, clique-width.

Support<sup>1</sup>

## 1 Introduction

Many polynomials associated with graphs, matroids or combinatorial maps count *configurations* in these objects. We take here "configuration" in a wide sense. Typical examples are colorings, matchings, stable subsets, subgraphs. In many cases, a *weight* is associated with the considered configurations : number of colors, cardinality, number of connected components or rank of an associated subgraph.

A *multivariate polynomial*, as the one defined by A. Sokal, that generalizes Tutte's two variable polynomial, not only counts configurations, but also enumerates them, together with their associated weights. Multivariate polynomials may have recursive definitions, the specializations of which give the classical recursive definitions for the corresponding polynomials. (Specialization is the inverse of generalization, these notions are explained below).

We think that a recursive definition at the multivariate level makes better understand what is going on at the usual level of one or two variable polynomials.

---

<sup>1</sup>This work has been supported by the ANR project GRAAL and by a temporary position of CNRS researcher.

Furthermore, multivariate polynomials may have *static definitions* based on properties of configurations and weights expressible in *second-order logic* or better in *monadic second-order logic*. Why better ? Because this yields fixed parameter tractable algorithms, where the parameters are the tree-width or the clique-width of a graph, or the branch-width of a matroid. This consequence has been explained in [CMR, Mak04, Mak05]. We apply these ideas to the *interlace polynomials* defined first under this name in [ABS], then generalized in [ABS04b] and [AvH] (and previously defined under another name, see Las Vergnas [LV]), that is, we give a common multivariate generalization and show that is is definable by monadic second-order formulas.

What do we mean by "generalize" ? If two polynomials  $P(G)$  and  $Q(G)$  are associated with each graph  $G$  of a certain type, we say that  $P(G)$  is *more general* than  $Q(G)$  and that  $Q(G)$  is a *specialization* of  $P(G)$  if  $Q(G)$  can be obtained from  $P(G)$  by a *substitution* of fixed polynomials to the variables of  $P(G)$ ; such a substitution may be also combined with multiplication by polynomials depending, say, on the number of vertices and/or edges. We do not try to propose here a most general definition. Applications of the idea will suffice.

A *multivariate polynomial* is one with indeterminates depending on the vertices or the edges of the considered graph (such indeterminates are sometimes called "weights", because they make possible to evaluate the polynomial with distinct values associated with distinct vertices or edges). Sokal's multivariate Tutte polynomial of a graph  $G = (V, E)$  is defined by :

$$Z(G) = \sum_{A \subseteq E} u^{k(G[A])} \prod_{e \in A} v_e$$

where  $G[A]$  is the subgraph of  $G$  with set of vertices  $V$  and set of edges  $A$ ,  $k(G[A])$  is the number of its connected components. This polynomial belongs to  $\mathbf{Z}[u, v_e ; e \in E]$ . An indeterminate  $v_e$  is associated with each edge  $e$ . The indeterminates commute, the order of enumeration over each set  $A$  is irrelevant. For two graphs  $G$  and  $G'$  with sets of edges in bijection, we have  $Z(G) = Z(G')$  (where the variables indexed by edges of  $G$  and  $G'$  correspond via the considered bijection) iff  $|V(G)| = |V(G')|$  and their cycle matroids are isomorphic (via the same bijection between edges). This observation explains what information is contained in this polynomial about the considered graph.

The polynomial  $Z(G)$  is more general than Tutte's two variable polynomial  $T(G, x, y)$  because (see [Sok] for details) :

$$T(G, x, y) = ((x - 1)^{k(G)} (y - 1)^{|V|})^{-1} \alpha(Z(G))$$

where  $\alpha$  is the substitution :

$$[u := (x - 1)(y - 1); v_e := y - 1 \text{ for all } e \in E].$$

Conversely, one can express  $Z'(G)$ , defined as  $Z(G)$  where every indeterminate  $v_e$  replaced by the same indeterminate  $v$ , in terms of  $T(G, x, y)$  in a similar way. Hence,  $Z'(G)$  and  $T(G)$  are equivalent both in expressive power and for the complexity of their computations.

In this article, we define a multivariate polynomial, that generalizes the different *interlace* polynomials defined for graphs in [ABS], [ABS04b] and [AvH], and also, the *independence* polynomial surveyed in [LM]. In our polynomial, a configuration is a set  $A$  of vertices and its associated value is the rank of the induced subgraph. By *rank* we mean, as in these articles, the rank of the adjacency matrix with respect to the field  $\text{GF}(2)$ . Actually, in order to generalize a polynomial defined in [AvH], we introduce a second argument in a configuration, namely a set  $B$  of vertices corresponding to "toggled loops" : if a vertex in  $B$  has no loop we add one ; if it has a loop, we delete it. Then we evaluate ranks.

We find for this polynomial a recursive definition, somewhat more complicated than the usual ones based on contracting and deleting edges. The polynomials of [ABS], [ABS04b] and [AvH] are specializations of ours, and we find their recursive definitions as the corresponding specializations, sometimes with the necessity of proving nontrivial properties.

This approach is based on *static definitions* from which *recursive definitions* can be constructed, and not the other way around. Here we follow Sokal who considers recursive definitions as secondary. It is developed in a fundamental perspective in [CGM]. Let us say to have a representative picture of the most general case that the polynomials we obtain in this way are of the form :

$$P(G) = \sum_{C \in \Gamma(G)} n_C \cdot v_C u^{f(C)}$$

where  $C$  ranges over all configurations of a multiset  $\Gamma(G)$ ,  $n_C$  is the number of occurrences of  $C$  in  $\Gamma(G)$ ,  $v_C$  is a monomial (like  $\prod_{e \in A} v_e$  in the above polynomial  $Z(G)$ ) that describes configuration  $C$ , and  $f(C)$  is the weight of  $C$ . Polynomials of this form have necessarily positive coefficients. We are especially interested in cases where  $\Gamma(G)$  and  $f$  can be expressed by monadic second-order formulas because of algorithmic applications developed in [CMR, Mak04, Mak05].

Such polynomials are usually of exponential size. We may ask for a polynomial evaluation of the *truncation* of  $P(G)$  i.e., of its part limited to the monomials associated with configurations of a given size. For such purpose, having a description of  $\Gamma(G)$  and of the weight function  $f$  by monadic second order formulas is helpful, as we will see in the last section, because this yields *fixed parameter tractable algorithms*, where the parameter is tree-width or clique-width.

Summary of the article : 2. Definitions and basic facts, 3. A multivariate interlace polynomial and its recursive definition, 4. Specializations to known polynomials, 5. Polynomial time evaluation for graphs of bounded clique-width, 6. Conclusion, 7. References, 8. Appendix : The Tutte polynomial

## 2 Definitions and basic facts

Graphs are finite, simple, undirected, possibly with loops. A graph is defined as a pair  $G = (V_G, A_G)$  of a set of vertices  $V_G$  and a symmetric adjacency matrix  $A_G$  over  $\text{GF}(2)$ . We omit the subscripts whenever possible without ambiguity. The *rank*  $rk(G)$  of  $G = (V, A)$  is defined as the rank  $rk(A)$  of  $A$  over  $\text{GF}(2)$ ; its *corank* (or *nullity*) is  $n(G) := n(A) := |V| - rk(A)$ . The empty graph  $\emptyset$  has rank and corank 0.

The set of looped vertices of  $G$  (the vertices  $i$  such that  $A(i, i) = 1$ ) is denoted by  $Loops(G)$ . For  $a$  in  $V$ , we let  $N(G, a)$  be the set of neighbours  $b$  of  $a$ , with  $b \neq a$ . (A looped vertex is not a neighbour of itself).

If  $X$  is a set of vertices, we let  $G - X$  denote  $G[V - X]$ , the induced subgraph of  $G$  with set of vertices  $V - X$ .

We denote by  $G \nabla X$  the graph obtained by "toggling" the loops in  $X$ , i.e.,  $V_{G \nabla X} := V_G$  and :

$$\begin{aligned} A_{G \nabla X}(i, j) &:= 1 - A_G(i, j) \quad \text{if } i = j \in X, \\ A_{G \nabla X}(i, j) &:= A_G(i, j) \quad \text{otherwise.} \end{aligned}$$

We write  $G = H \oplus K$  if  $G$  is the union of disjoint subgraphs  $H$  and  $K$ .

For two graphs  $G$  and  $H$  we write  $H = h(G)$  and we say that they are *isomorphic by  $h$*  if  $h$  is a bijection of  $V_G$  onto  $V_H$  and  $A_H(h(i), h(j)) = A_G(i, j)$  for all  $i$  and  $j$ .

### *Pivoting and local complementation*

We recall the precise definitions of operations like local complementation and pivoting, because there are some variants in articles.

For  $a$  and  $b$  distinct vertices of  $G$  we define the graph  $H = G^{ab}$  as follows :

$$\begin{aligned} V_H &:= V_G \text{ and} \\ A_H(i, j) &:= 1 - A_G(i, j) \quad \text{if the following holds :} \\ \{i, j\} \cap \{a, b\} &= \emptyset \text{ and} \\ &\{\text{either } i \in N(G, a) - N(G, b) \text{ and } j \in N(G, b), \\ &\text{or } j \in N(G, a) - N(G, b) \text{ and } i \in N(G, b), \\ &\text{or } i \in N(G, b) - N(G, a) \text{ and } j \in N(G, a), \\ &\text{or } j \in N(G, b) - N(G, a) \text{ and } i \in N(G, a)\}. \end{aligned}$$

In all other cases, we let  $A_H(i, j) := A_G(i, j)$ .

This operation is called *pivoting* on  $a, b$ . It does not depend on whether  $a$  and  $b$  are loops or are adjacent.

*Local complementation* : for a vertex  $a$  of  $G$  we define  $H = G^a$  as follows :

$$\begin{aligned} V_H &:= V_G \text{ and :} \\ A_H(i, j) &:= 1 - A_G(i, j) \quad \text{if } i, j \in N(G, a), \text{ including the case } i = j. \end{aligned}$$

$A_H(i, j) := A_G(i, j)$  otherwise.

Another notion of local complementation is defined by :

$$G * a = (G \nabla N(G, a))^a = G^a \nabla N(G, a).$$

It "toggles" the (non loop) edges of  $G[N(G, a)]$ . It is used for graphs without loops in the characterization of circle graphs and in the definition of vertex-minors. ([Bou], [Oum], [CouOum]).

We write  $a - b$  to express that  $a$  and  $b$  are adjacent both without loops, and  $a^\ell - b$  to express the same with  $a$  looped and  $b$  not looped, and  $a^\ell - b^\ell$  if  $a$  and  $b$  are looped. These operations satisfy properties listed in the following lemma :

**Lemma 1** : For every graph  $G = (V, A)$ , for distinct vertices  $a, b$  and all sets of vertices  $X, Y$  we have :

- (1)  $(G^a)^a = G$ ;  $G^{ab} = G^{ba}$  ;  $(G^{ab})^{ab} = G$  ;
- (2)  $G^{ab} = h(((G^a)^b)^a \nabla a)$  if  $b \in N(G, a)$  and  $h$  is the permutation of  $V$  that exchanges  $a$  and  $b$  ;  $G^{ab} - a - b = ((G^a)^b)^a - a - b$ .
- (3)  $(G^{ab})^b = h(((G^a)^b)^a \nabla a)$  if  $a, b, h$  are as in (2) ;  $(G^{ab})^b - a - b = (G^a)^b - a - b$ .
- (4)  $G \nabla X^{ab} = G^{ab} \nabla X$  ;  $G \nabla X^a = G^a \nabla X$  ;  $G \nabla X[Y] = G[Y] \nabla (X \cap Y)$ .
- (5)  $G[X]^{ab} = G^{ab}[X]$  ;  $G[X]^a = G^a[X]$  if  $a$  and  $b$  are not in  $X$ .

**Proof** : (1), (4), (5) are clear from the definitions.

(2) is a well-known fact about pivoting and local complementation. See for instance [ABS04b].

(3) This is a consequence of (1), (2) and (4) :

$$\begin{aligned} (G^{ab})^b &= (h(((G^a)^b)^a \nabla a))^b \\ &= h(((G^a)^b)^a \nabla a)^a = h((((G^a)^b)^a)^a \nabla a) = h((G^a)^b \nabla a). \square \end{aligned}$$

*Computating ranks of graphs.*

**Lemma 2** : For every graph  $G$ , for distinct vertices  $a, b$  we have :

- (1)  $rk(G) = 1 + rk(G^a - a)$  if  $a \in Loops(G)$  ;
- (2)  $rk(G) = 2 + rk(G^{ab} - a - b)$  if  $a - b$  ;
- (3)  $rk(G - a) = rk(G^{ab} - a)$  if  $a - b$  ;
- (4)  $rk(G) = 2 + rk((G^a)^b - a - b) = 1 + rk(G^{ab} - b)$  if  $a^\ell - b$  .

**Proof** : (1)-(3) are proved in [ABS04b].

(4) We note that  $(G^a)^b - a - b = (G^a - a)^b - b$  and that  $(G^a - a)$  has a loop on  $b$ . Hence by using (1) twice :

$$rk((G^a)^b - a - b) = rk((G^a - a)^b - b) = rk(G^a - a) - 1 = rk(G) - 2.$$

For the second equality :

$$\begin{aligned}
rk(G) &= rk((G^b)^a - a - b) + 2 = rk((((G^b)^a)^b - a - b) + 2 \\
&= rk((((G^b)^a)^b - a)^b - b) + 2 = rk(((G^b)^a)^b - a) + 1 \\
&\text{because } ((G^b)^a)^b - a \text{ has a loop on } b, \text{ hence :} \\
rk(G) &= rk(h(G^{ba}\nabla b) - a) + 1 = rk(h(G^{ba}\nabla b - b)) + 1 \\
&= rk(G^{ba}\nabla b - b) + 1 = rk(G^{ab} - b) + 1 \\
&\text{because } G^{ba}\nabla b - b = G^{ab} - b. \square
\end{aligned}$$

### 3 The multivariate interlace polynomial

Polynomials have integer coefficients. Following Sokal [Sok] we call *multivariate polynomials* those with indeterminates  $x_a, y_a, z_a, \dots$  associated with vertices  $a$  of the considered graph  $G$ . We will denote by  $\mathbf{X}_G$  the set of such indeterminates for  $X = \{x, y, z, \dots\}$ . They are the *G-indexed indeterminates*. We denote by  $U$  a set  $\{u, v, w, \dots\}$  of "ordinary" indeterminates not associated with elements of graphs.

By a *polynomial*  $P(G)$ , we mean a mapping  $P$  that associates with a graph  $G$  a polynomial in  $\mathbf{Z}[U \cup \mathbf{X}_G]$  such that if  $h$  is an isomorphism of  $G$  onto  $H$ , then  $P(H)$  is obtained from  $P(G)$  by the substitution that replaces  $x_a$  by  $x_{h(a)}$  for every  $x_a$  in  $\mathbf{X}_G$ .

A *specializing substitution* is a substitution that replaces an indeterminate from a finite set  $U = \{u, v, w, \dots\}$  by a polynomial in  $\mathbf{Z}[U]$ , and a  $G$ -indexed indeterminate  $x_a$  in  $\mathbf{X}_G$ , by a polynomial in  $\mathbf{Z}[U \cup \{y_a \mid y \in X\}]$ , the same for each  $a$ . For an example, such a substitution can replace  $x_a$  by  $y_a(x-1)^2 - 3z_a u + 1$ , for every vertex  $a$  of every graph. If  $\sigma$  is a specializing substitution, then  $\sigma \circ P$ , defined by  $\sigma \circ P(G) = \sigma(P(G))$  is in this sense a polynomial.

For a set  $A$  of vertices we let  $x_A$  abbreviate the product (in any order) of the commutative indeterminates  $x_a$ , for  $a$  in  $A$ . If  $A = \emptyset$ , then  $x_A = 1$ . If  $B$  is a set of subsets of  $G$ , then the polynomial  $\sum_{A \in B} x_A$  describes exactly  $B$ . If  $B$  is a multiset of sets, then it is described by  $\sum_{A \in B} n(A) \cdot x_A$  where  $n(A)$  is the number of occurrences of  $A$  in  $B$ .

**Definition 3 :** *The multivariate interlace polynomial.*

For a graph  $G$  we define

$$B(G) = \sum_{A \cap B = \emptyset} x_A y_B u^{rk(G \nabla B[A \cup B])} v^{n(G \nabla B[A \cup B])}$$

where  $A, B$  range over subsets of  $V$ . Hence  $B(G) \in \mathbf{Z}[\{u, v\} \cup \mathbf{X}_G]$  where  $X = \{x, y\}$ .

The interlace polynomial  $q$  of [ABS04b] is obtained from  $B(G)$  by a substitution:  $q(G; x, y) = \sigma(B(G))$  where  $\sigma$  is the substitution :

$$[u := x - 1; v := y - 1; x_a := 1, y_a := 0 \text{ for all } a \in V],$$

and the polynomial  $Q$  of [AvH], defined for graphs without loops is  $Q(G, x) = \tau(B(G))$  where  $\tau$  is the substitution

$$[u := 1; v := x - 2; x_a := y_a := 1 \text{ for all } a \in V].$$

These polynomials are actually *defined recursively* in these articles (which raises a problem of well-definedness), and then proved to be equal to the polynomials  $\sigma(B(G))$  and  $\tau(B(G))$ .

For more clarity with variable names, we will write  $q(G)$  and  $Q(G)$  with variables  $u'$  and  $v'$  instead of  $x$  and  $y$ . Hence  $q(G; u', v') = \sigma(B(G))$  where  $\sigma$  is the substitution :

$$[u := u' - 1; v := v' - 1; x_a := 1, y_a := 0 \text{ for all } a \in V],$$

and  $Q(G, v') = \tau(B(G))$  where  $\tau$  is the substitution

$$[u := 1; v := v' - 2; x_a := y_a := 1 \text{ for all } a \in V].$$

Let  $B_1(G)$  be the polynomial obtained from  $B(G)$  by replacing  $v$  by 1.

**Lemma 4 :** For every graph  $G$  and every set  $T$  of vertices :

- (1)  $B(G) = \theta(B_1(G))$  where :  
 $\theta := [u := uv^{-1}; x_a := vx_a; y_a := vy_a \text{ for all } a \in V],$
- (2)  $B(G \nabla T) = \mu(B(G))$  where :  
 $\mu := [x_a := y_a, y_a := x_a \text{ for all } a \in T].$

**Proof :** (1) Clear.

(2) We observe that  $G \nabla T \nabla B[A \cup B] = G \nabla (A' \cup B')[A \cup B]$  where  $A' = A \cap T, B' = B - B \cap T$ . The result follows.  $\square$

We will write :  $B = \theta \circ B_1$ . The polynomial  $B(G)$  can thus be "recovered" from  $B_1(G)$ . Since every graph  $G$  is  $G_1 \nabla T$  for some  $T$  with  $G_1$  without loops, we have  $B(G) = \mu(B(G_1))$  where  $\mu$  is as in Lemma 4. Hence, it is enough to know  $B(G)$  for graphs  $G$  without loops. However, the recursive definitions to be considered below will introduce graphs with loops in the recursive calls.

#### *Properties of polynomials*

The polynomials  $q$  and  $Q$  defined above satisfy some properties for all graphs  $G$  :

$$q(G - a) - q(G - a - b) = q(G^{ab} - a) - q(G^{ab} - a - b) \text{ if } a - b(1)$$

$$Q(G * a) = Q(G) \quad (2)$$

$$Q(G^{ab}) = Q(G). \quad (3)$$

Do these equalities hold for  $B(G)$  ? The answer is no for (2) and (3) as a consequence of the next proposition (and also for (1), see below Counter-example 14).

**Proposition 5 :** A graph  $G$  and its polynomial  $B(G)$  can be reconstructed from  $\rho(B(G))$  where  $\rho := [v := 1; y_a := 0 \text{ for all } a \in V]$ .

**Proof :** For every set of vertices  $A$ , the rank of  $G[A]$  is the unique integer  $n$  such that  $x_A u^n$  is a monomial of  $\rho(B(G))$ . Now a vertex  $a$  has a loop if  $rk(G[a]) = 1$ , and no loop if  $rk(G[a]) = 0$ . Hence, we obtain  $Loops(G)$  from  $\rho(B(G))$ . Using this information, we can reconstruct edges.

If  $a$  and  $b$  are not looped, they are adjacent iff  $rk(G[\{a, b\}]) = 2$ , otherwise  $rk(G[\{a, b\}]) = 0$ .

If one of  $a, b$  is looped, they are adjacent iff  $rk(G[\{a, b\}]) = 2$ , otherwise  $rk(G[\{a, b\}]) = 1$ .

If both are looped, they are adjacent iff  $rk(G[\{a, b\}]) = 1$ , otherwise  $rk(G[\{a, b\}]) = 2$ .  $\square$

This proof shows how strange is the behaviour of the rank function on graphs. It follows that identities (2) and (3) cannot hold for  $B$  and even for  $\rho \circ B$ .

**Question :** By which algebraic transformations can  $B(G)$  be expressed in terms of  $\rho(B(G))$ , in a uniform way, valid for all graphs  $G$  ?

### 3.1 Recursive definition

We now determine a recursive definition, (also called a set of *reduction formulas*) of  $B(G)$  from which will follow the recursive definitions of [ABS] and [AvH].

We let  $a$  denote the graph with one non-looped vertex  $a$ , and  $a^\ell$  the similar graph with looped vertex  $a$ .

**Lemma 6 :** For every graph  $G$ , for every graph  $H$  disjoint from  $G$  we have:

- (1)  $B(\emptyset) = 1$
- (2)  $B(G \oplus H) = B(G) \cdot B(H)$
- (3)  $B(a) = 1 + x_a v + y_a u$
- (4)  $B(a^\ell) = 1 + x_a u + y_a v$ .

**Proof :** Easy verification from the definitions.  $\square$

The more complicated task consists now in expressing  $B(G)$  in the case where  $a$  and  $b$  are adjacent (this is necessary if no rule of Lemma 6 is applicable). We will distinguish three cases :  $a - b$ ,  $a^\ell - b$ , and  $a^\ell - b^\ell$ .

For a graph  $G$  and disjoint sets of vertices  $A$  and  $B$ , we let  $m(G, A, B)$  denote the monomial  $x_A y_B u^{rk(G \nabla B[A \cup B])} v^{n(G \nabla B[A \cup B])}$  so that  $B(G)$  is nothing but the sum of them over all pairs  $A, B$  (the condition  $A \cap B = \emptyset$  will be assumed for each use of the notation  $m(G, A, B)$ ).

For distinct vertices  $a, b$ , two disjoint sets  $A, B$  can contain  $a, b$  or not according to 9 cases. We let  $i \in \{0, 1, 2\}$  mean that a vertex is in  $V - (A \cup B)$ , in  $A$  or in  $B$  respectively. Let  $B_{ij}$  be the sum of monomials  $m(G, A, B)$  such that  $i$  tells where is  $a$ , and  $j$  tells where is  $b$ . For an example :  $B_{02}$  is the sum of monomials  $m(G, A, B)$  such that  $a \in V - (A \cup B)$ ,  $b \in B$ .

**Claim 7** : Let  $G$  with  $a - b$ .

- (1)  $B_{00} = B(G - a - b)$
- (2)  $B_{11} = x_a x_b u^2 \cdot B(G^{ab} - a - b)$ .
- (3)  $B_{20} = y_a u \cdot B(G^a - a - b)$  ;  $B_{02} = y_b u \cdot B(G^b - a - b)$  ;
- (4)  $B_{12} = x_a y_b u^2 \cdot B((G^b)^a - a - b)$  ;  $B_{21} = x_b y_a u^2 \cdot B((G^a)^b - a - b)$ .

**Proof** : (1) Clear from the definitions.

(2) A monomial of  $B_{11}$  is of the form :

$$m(G, A, B) = x_A y_B u^{rk(G \nabla B[A \cup B])} v^{n(G \nabla B[A \cup B])} \quad (1)$$

with  $a, b \in A$  (because of the subscript 11). By Lemma 2(2) we have :

$$rk(G \nabla B[A \cup B]) = 2 + rk(G \nabla B[A \cup B]^{ab} - a - b).$$

But  $G \nabla B[A \cup B]^{ab} - a - b = (G^{ab} - a - b) \nabla B[A' \cup B]$  where  $A' = A - a - b$  (we use here Lemma 1(4,5)). Hence :

$$m(G, A, B) = x_a x_b u^2 \cdot m(G^{ab} - a - b, A', B).$$

It follows that :

$$B_{11} = x_a x_b u^2 \cdot B(G^{ab} - a - b)$$

because the set of pairs  $A', B \subseteq V - a - b$  such that  $A'$  and  $B$  are disjoint coincides with the set of pairs  $(A - a - b), B$  such that  $A, B \subseteq V$ ,  $A$  and  $B$  are disjoint subsets of  $V$  and  $a, b \in A$ .

(3) The proof is similar. A monomial of  $B_{20}$  is of the form (1) above with  $a \in B$ ,  $b \notin A \cup B$  (because of the subscript 20). By Lemma 2(1) we have:

$$rk(G \nabla B[A \cup B]) = 1 + rk(G \nabla B[A \cup B]^a - a)$$

because  $a$  is looped in  $G \nabla B[A \cup B]$ . But :

$$G \nabla B[A \cup B]^a - a = (G \nabla a^a - a - b) \nabla B'[A \cup B']$$

because  $b \notin A \cup B$ , where  $B' = B - a$ . (By Lemma 1). Clearly,  $(G\nabla a^a - a - b) = (G^a - a - b)$ . Hence  $m(G, A, B) = y_a u \cdot m(G^a - a - b, A, B')$ . It follows that :

$$B_{20} = y_a u \cdot B(G^a - a - b)$$

because the set of pairs  $A, B' \subseteq V - a - b$  such that  $A$  and  $B'$  are disjoint coincides with the set of pairs  $A, (B - a)$  such that  $A, B \subseteq V$ ,  $A$  and  $B$  are disjoint subsets of  $V$ ,  $a \in B$  and  $b \notin A \cup B$ . The case of  $B_{02}$  is obtained by exchanging  $a$  and  $b$ .

(4) A monomial of  $B_{12}$  is of the form (1) above with  $a \in A$ ,  $b \in B$ . By Lemma 2(4) we have :

$$rk(G\nabla B[A \cup B]) = 2 + rk((G\nabla B[A \cup B]^b)^a - a - b)$$

because  $b^b - a$  in  $G\nabla B[A \cup B]$ . We have :

$$(G\nabla B[A \cup B]^b)^a - a - b = ((G^b)^a - a - b) \nabla B'[A' \cup B']$$

where  $A' = A - a$ ,  $B' = B - b$ . Hence :

$$m(G, A, B) = x_a y_b u^2 \cdot m((G^b)^a - a - b, A', B').$$

It follows that :

$$B_{12} = x_a y_b u^2 \cdot B((G^b)^a - a - b)$$

because the set of pairs  $A', B' \subseteq V - a - b$  such that  $A'$  and  $B'$  are disjoint coincides with the set of pairs  $(A - a), (B - b)$  such that  $A, B \subseteq V$ ,  $A$  and  $B$  are disjoint  $a \in A$  and  $b \in B$ . The case of  $B_{21}$  is obtained similarly by exchanging  $a$  and  $b$ .  $\square$

The next claim establishes linear relations between some polynomials  $B_{ij}$ .

**Claim 8 :** Let  $G$  with  $a - b$ .

- (1)  $B(G - a) = B_{00} + B_{01} + B_{02}$
- (2)  $B(G - b) = B_{00} + B_{10} + B_{20}$
- (3)  $u y_a \cdot B(G^a - a) = B_{20} + B_{21} + B_{22}$
- (4)  $u y_b \cdot B(G^b - b) = B_{02} + B_{12} + B_{22}$

**Proof :** (1), (2) Clear from the definitions.

(3) From the definitions,  $B_{20} + B_{21} + B_{22}$  is the sum of monomials  $m(G, A, B)$  such that  $a \in B$ . We have :

$$rk(G\nabla B[A \cup B]) = 1 + rk(G\nabla B[A \cup B]^a - a)$$

by Lemma 2(1). But :

$$\begin{aligned} G\nabla B[A \cup B]^a - a &= ((G\nabla a)^a - a)\nabla B'[A \cup B'] \quad (\text{where } B' = B - a) \\ &= (G^a - a)\nabla B'[A \cup B']. \end{aligned}$$

This gives the result with the usual argument.

(4) Similar to (3) by exchanging  $a$  and  $b$ .  $\square$

If we collect the equalities of Claims 7 and 8 we have 10 definitions or linear equalities for 9 "unknowns". This is enough for obtaining  $B(G)$ . We get thus :

$$\begin{aligned} B(G) &= (B_{00} + B_{10} + B_{20}) + \{B_{01} + B_{11} + B_{21}\} + (B_{02} + B_{12} + B_{22}) \\ &= B(G - b) + \{B_{01} + x_a x_b u^2 \cdot B(G^{ab} - a - b) + x_b y_a u^2 \cdot B((G^a)^b - a - b)\} + \\ &\quad + y_b u \cdot B(G^b - b). \end{aligned}$$

$$\begin{aligned} \text{Then } B_{01} &= B(G - a) - B_{00} - B_{02} \\ &= B(G - a) - B(G - a - b) - y_b u \cdot B(G^b - a - b). \end{aligned}$$

We obtain, after reorganization of the expression and a bit of factorization :

**Lemma 9** : Let  $G$  with  $a - b$ . We have :

$$\begin{aligned} B(G) &= x_b u^2 \{x_a \cdot B(G^{ab} - a - b) + y_a \cdot B((G^a)^b - a - b)\} + \\ &\quad + y_b u \{B(G^b - b) - B(G^b - a - b)\} + \\ &\quad + B(G - a) + B(G - b) - B(G - a - b). \end{aligned}$$

Considering  $B_{22}$  for which we have two expressions, we get :

**Corollary 10** : Let  $G$  with  $a - b$ .

$$\begin{aligned} &y_b \{B(G^b - b) - B(G^b - a - b) - x_a u \cdot B((G^a)^b - a - b)\} \\ &= y_a \{B(G^a - a) - B(G^a - a - b) - x_b u \cdot B((G^b)^a - a - b)\}. \end{aligned}$$

Next we consider the cases  $a^\ell - b$  and  $a^\ell - b^\ell$ . Actually, Lemma 4(2) will shorten the computations.

**Lemma 11** : (1) Let  $G$  with  $a - b^\ell$ .

$$\begin{aligned} B(G) &= y_b u^2 \{x_a \cdot B(G^{ab} - a - b) + y_a \cdot B((G^a)^b - a - b)\} \\ &\quad + x_b u \{B(G^b - b) - B(G^b - a - b)\} \\ &\quad + B(G - a) + B(G - b) - B(G - a - b). \end{aligned}$$

(2) Let  $G$  with  $a^\ell - b^\ell$ .

$$\begin{aligned} B(G) &= y_b u^2 \{y_a \cdot B(G^{ab} - a - b) + x_a \cdot B((G^a)^b - a - b)\} \\ &\quad + x_b u \{B(G^b - b) - B(G^b - a - b)\} \\ &\quad + B(G - a) + B(G - b) - B(G - a - b). \end{aligned}$$

**Proof** : (1) We have  $G = G_1 \nabla b$ ,  $G_1 = G \nabla b$ , where in  $G_1$  we have  $a - b$  so that Lemma 9 is applicable.

We get then, letting  $\beta$  be the substitution that exchanges  $x_b$  and  $y_b$  :

$$B(G) = \beta(B(G_1)) =$$

$$\begin{aligned}
&= y_b u^2 \{x_a \cdot B(G\nabla b^{ab} - a - b) + y_a \cdot B((G\nabla b^a)^b - a - b)\} + \\
&\quad + x_b u \{B(G\nabla b^b - b) - B(G\nabla b^b - a - b)\} + \\
&\quad + \beta(B(G\nabla b - a)) + B(G\nabla b - b) - B(G\nabla b - a - b) \\
&= y_b u^2 \{x_a \cdot B(G^{ab} - a - b) + y_a \cdot B((G^a)^b - a - b)\} \\
&\quad + x_b u \{B(G^b - b) - B(G^b - a - b)\} \\
&\quad + B(G - a) + B(G - b) - B(G - a - b),
\end{aligned}$$

because we have  $G\nabla b^{ab} - a - b = G^{ab} - a - b$  and  $B(G\nabla b^{ab} - a - b)$  has no occurrence of an indeterminate indexed by  $b$ ,

because we have  $(G\nabla b^a)^b - a - b = (G^a)^b - a - b$  and  $B((G\nabla b^a)^b - a - b)$  has no occurrence of an indeterminate indexed by  $b$ ,

and similar remarks apply to  $G\nabla b^b - b$ , to  $G\nabla b^b - a - b$ , to  $G\nabla b - b$ , and to  $G\nabla b - a - b$ . Finally, we have  $\beta(B(G\nabla b - a)) = B(G - a)$  by Lemma 1 and Lemma 4.

(2) Very similar argument.  $\square$

We can now sum up the results of lemmas 6, 9, 11 into the following proposition, where the three cases are collected into a single one with help of the little trick of introducing "metavariables"  $z_c, w_c$  for each  $c \in V$  :

$$\begin{aligned}
z_c &= x_c \text{ and } w_c = y_c \text{ if } c \text{ is not a loop,} \\
z_c &= y_c \text{ and } w_c = x_c \text{ if } c \text{ is a loop.}
\end{aligned}$$

**Proposition 12** : For every graph  $G$ , for every graph  $H$  disjoint from  $G$ , every vertex  $a$ , we have :

- (1)  $B(\emptyset) = 1$
- (2)  $B(G \oplus H) = B(G) \cdot B(H)$
- (3)  $B(a) = 1 + x_a v + y_a u$
- (4)  $B(a^t) = 1 + x_a u + y_a v$
- (5)  $B(G) = z_b u^2 \{z_a \cdot B(G^{ab} - a - b) + w_a \cdot B((G^a)^b - a - b)\} \\ + w_b u \{B(G^b - b) - B(G^b - a - b)\} \\ + B(G - a) + B(G - b) - B(G - a - b).$

if  $b \in N(G, a)$ .

**Proof** : Immediate consequence of Lemmas 6,9,11.  $\square$

We have an even shorter definition :

**Corollary 13** : For every graph  $G$  every vertex  $a$ , we have :

- (1)  $B(\emptyset) = 1$
- (2)  $B(G) = (1 + z_a v + w_a u)B(G - a)$  if  $N(G, a) = \emptyset$ ,
- (3)  $B(G) = z_b u^2 \{z_a B(G^{ab} - a - b) + w_a B((G^a)^b - a - b)\} \\ + w_b u \{B(G^b - b) - B(G^b - a - b)\} \\ + B(G - a) + B(G - b) - B(G - a - b).$

if  $b \in N(G, a)$ .

**Counter-example 14** :

It is proved in Proposition 8 of [ABS04b] that if  $a - b$  in  $G$  then :

$$q(G - a) - q(G - a - b) = q(G^{ab} - a) - q(G^{ab} - a - b)$$

This is not true for  $B$ . Take  $G = c - a - b - d$ . Note that  $G^{ab}$  is  $G$  augmented with  $c - d$ . Assume we would have :

$$B(G - a) - B(G - a - b) = B(G^{ab} - a) - B(G^{ab} - a - b). \quad (*)$$

In the left handside, we have a single monomial of the form  $y_b y_c x_d u^n$  for some  $n$ , and it must be from  $B(G - a)$  because  $b$  is not in  $G - a - b$ . This monomial is  $y_b y_c x_d u^3$  because  $rk(c^\ell \oplus (d - b^\ell)) = 3$ . In the right handside we have the monomial  $y_b y_c x_d u^2$  because  $rk(c^\ell - d - b^\ell) = 2$ . Hence we cannot have Equality (\*).□

In such a case, we can ask what is the less specialized (or most general) substitution  $\sigma$  such that the corresponding equality is true for  $\sigma \circ B$  ? Some answers will be given below. We prove actually a more complicated identity.

**Proposition 15** : If  $a - b$  in  $G$  then :

$$B(G - a) - B(G - a - b) - B(G^{ab} - a) + B(G^{ab} - a - b) = y_b u \{ B(G^b - a - b) - B((G^a)^b - a - b) \}.$$

**Proof** : We use the notation and some facts from Claims 7 and 8 :

$$B(G - a) - B(G - a - b) = B_{01} + B_{02} = B_{01} + y_b u \cdot B(G^b - a - b).$$

We let  $B_{01}^{ab}$  and  $B_{02}^{ab}$  denote the polynomials  $B_{01}$  and  $B_{02}$  relative to  $(G^{ab}, a, b)$  instead of to  $(G, a, b)$ . Then we have :

$$B(G^{ab} - a) - B(G^{ab} - a - b) = B_{01}^{ab} + B_{02}^{ab} = B_{01}^{ab} + y_b u \cdot B((G^{ab})^b - a - b).$$

We have by Lemma 1 :  $(G^{ab})^b - a - b = (G^a)^b - a - b$ .

On the other hand,  $B_{01}^{ab}$  is the sum of monomials :

$$m(G^{ab}, A, B) = x_A y_B u^{rk(G^{ab} \nabla B[A \cup B])} v^{n(G^{ab} \nabla B[A \cup B])}$$

for disjoint sets  $A, B$  such that  $a \notin A \cup B, b \in A$ . But for such  $A, B$  :

$$G^{ab} \nabla B[A \cup B] = G \nabla B[A \cup B \cup a]^{ab} - a.$$

Hence, using Lemma 1 and Lemma 2(3) :

$$\begin{aligned} rk(G^{ab} \nabla B[A \cup B]) &= rk(G \nabla B[A \cup B \cup a]^{ab} - a) \\ &= rk(G \nabla B[A \cup B \cup a] - a) \\ &= rk(G \nabla B[A \cup B]). \end{aligned}$$

We have also  $n(G^{ab} \nabla B[A \cup B]) = n(G \nabla B[A \cup B])$ . Hence,  $m(G^{ab}, A, B) = m(G, A, B)$  and  $B_{01}^{ab} = B_{01}$ .

Collecting these remarks we get :

$$\begin{aligned}
& B(G - a) - B(G - a - b) - B(G^{ab} - a) + B(G^{ab} - a - b) \\
& = B_{02} - B_{02}^{ab} \\
& = y_b u \cdot B(G^b - a - b) - y_b u \cdot B((G^a)^b - a - b). \square
\end{aligned}$$

We note for later use that Identity (\*) of Counter-example 14 holds if either  $u = 0$  or  $y_b = 0$  for all  $b$ .

A polynomial  $P$  in  $\mathbf{Z}[X]$  is said to be *positive* if the coefficients of its monomials are positive. A polynomial in the sense of a mapping from graphs to polynomials is *positive* if  $P(G)$  is positive for every graph  $G$ . It is clear from Definition 3 that  $B$  is positive. This not immediate from the recursive definition of Corollary 13 because of two substractions in the right handside of (3). However, one can derive from Corollary 13 a stronger statement that is not immediate from Definition 3.

**Proposition 15 a :** For every graph  $G$  and every vertex  $a$ , the polynomials  $B(G)$  and  $B(G) - B(G - a)$  are positive.

**Proof :** By induction on the number of vertices of  $G$ , one proves simultaneously these two assertions by using Corollary 13.

In case (2) we have:

$$\begin{aligned}
B(G) - B(G - a) &= (1 + z_a v + w_a u) B(G - a) \text{ and in case (3) we have} \\
B(G) - B(G - a) &= z_b u^2 \{ z_a \cdot B(G^{ab} - a - b) + w_a \cdot B((G^a)^b - a - b) \} \\
&\quad + w_b u \{ B(G^b - b) - B(G^b - b - a) \} \\
&\quad + B(G - b) - B(G - b - a),
\end{aligned}$$

which gives with the induction hypothesis that  $B(G) - B(G - a)$  is positive. So is  $B(G)$  since, again by induction,  $B(G - a)$  is positive.  $\square$

## 4 Specializations to known polynomials

We have already observed that the polynomial  $q$  of [ABS04b] is  $q(G; u', v') = \sigma(B(G))$  where  $\sigma$  is the substitution :

$$[u := u' - 1; v := v' - 1; x_a := 1, y_a := 0 \text{ for all } a \in V],$$

and that the polynomial  $Q$  of [AvH], defined for graphs without loops is  $Q(G, v') = \tau(B(G))$  where  $\tau$  is the substitution

$$[u := 1; v := v' - 2; x_a := y_a := 1 \text{ for all } a \in V].$$

Both are actually specializations of the following two. We let :

$$B_{y=0}(G) := \sigma_0(B(G)) \text{ where } \sigma_0 \text{ is the substitution } [y_a := 0 \text{ for all } a \in V],$$

and

$B_{x=y}(G) := \sigma_=(B(G))$  where  $\sigma_ =$  is the substitution  $[y_a := x_a \text{ for all } a \in V]$ .

Polynomials  $B, B_{x=y}, B_{y=0}$  are by definition positive. This is also the case of  $Q$ , but this is not obvious from the above definitions. This fact follows for  $Q$  from the recursive definition established in [AvH] that we will reprove in a different way, but it does not from the definitions given in [ABS04b]. (In the appendix we give a proof that a multivariate version of the Tutte polynomial is positive.)

## 4.1 Fixed loops

The polynomial  $B_{y=0}(G)$  corresponds to "fixed loops" : it does not describe what happens when some loops are "toggled". It can be rewritten :

$$B_{y=0}(G) = \sum x_A u^{rk(G[A])} v^{n(G[A])}.$$

Clearly  $q(G; u', v') = \sigma'(B_{y=0}(G))$  where  $\sigma'$  is the substitution :

$$[u := u' - 1; v := v' - 1; x_a := 1 \text{ for all } a \in V].$$

**Proposition 16** : For every graph  $G$  every vertex  $a$ , we have :

- (1)  $B_{y=0}(\emptyset) = 1$
- (2)  $B_{y=0}(G) = (1 + x_a v) B_{y=0}(G - a)$  if  $N(G, a) = \emptyset$  and  $a$  is not a loop,
- (3)  $B_{y=0}(G) = x_a u \cdot B_{y=0}(G^a - a) + B_{y=0}(G - a)$  if  $a$  is a loop, isolated or not,
- (4)  $B_{y=0}(G) = x_b x_a u^2 \cdot B_{y=0}(G^{ab} - a - b) + B_{y=0}(G - a) + B_{y=0}(G - b) - B_{y=0}(G - a - b)$  if  $a - b$ .

**Proof** : (1), (2), (4) : Immediate from Corollary 13.

(3) If  $a$  is isolated, this follows from Corollary 13 (2). Otherwise, using the notation of the proof of Claim 7 we note that  $B_{y=0}(G)$  is the sum of monomials  $m(G, A, \emptyset)$  ; those such that  $a \notin A$  yield  $B(G - a)$ , the others yield  $x_a u \cdot B_{y=0}(G^a - a)$  since :

$$rk(G[A]) = rk(G[A]^a - a) + 1 = rk((G^a - a)[A - a]) + 1$$

by Lemma 2(1). This gives the result, however, it is interesting to see what gives Lemma 11. The two cases  $a^\ell - b$  and  $a^\ell - b^\ell$  yield the same equality.

$$B_{y=0}(G) = x_a u \{ B_{y=0}(G^a - a) - B_{y=0}(G^a - a - b) \} + B_{y=0}(G - a) + B_{y=0}(G - b) - B_{y=0}(G - a - b).$$

Hence we have to check that :

$$x_a u \cdot B_{y=0}(G^a - a - b) = B_{y=0}(G - b) - B_{y=0}(G - a - b).$$

This is nothing but Assertion (3) applied to  $H = G - b$ . Hence (3) can be established by induction on the size of  $G$ , with help of Lemma 11, and without repeating the analysis of the monomials  $m(G, A, \emptyset)$ .  $\square$

This proposition yields, with easy transformations, the following recursive definition of  $q$  :

- (q1)  $q(G) = v'^n$  if  $G$  consists of  $n$  isolated non-looped vertices,
- (q2)  $q(G) = (u' - 1)q(G^a - a) + q(G - a)$  if  $a$  is a loop, isolated or not,
- (q3)  $q(G) = (u' - 1)^2 q(G^{ab} - a - b) +$   
 $+q(G - a) + q(G - b) - q(G - a - b)$  if  $a - b$

which is not the one of [ABS04b] Proposition 6. The definition of this article replaces (q3) by :

- (q3')  $q(G) = ((u' - 1)^2 - 1)q(G^{ab} - a - b) +$   
 $+q(G - a) + q(G^{ab} - b)$  if  $a - b$

However, we have the following corollary of Proposition 15 that generalizes Proposition 8 of [ABS04b] :

**Corollary 17 :** If  $a - b$  in  $G$  then :

$$B_{y=0}(G - a) - B_{y=0}(G - a - b) = B_{y=0}(G^{ab} - a) - B_{y=0}(G^{ab} - a - b).$$

**Proof :** Immediate from Proposition 15 since  $y_b = 0$  for all  $b$ .  $\square$

We get thus the following corollary, and (q3') is equivalent to (q3).

**Corollary 18 :** For every graph  $G$  every vertex  $a$ , we have :

- (1)  $B_{y=0}(\emptyset) = 1$
- (2)  $B_{y=0}(G) = (1 + x_a v)B_{y=0}(G - a)$  if  $N(G, a) = \emptyset$  and  $a$  is not a loop,
- (3)  $B_{y=0}(G) = x_a u \cdot B_{y=0}(G^a - a) + B_{y=0}(G - a)$  if  $a$  is a loop, isolated or not,
- (4)  $B_{y=0}(G) = (x_b x_a u^2 - 1)B_{y=0}(G^{ab} - a - b) +$   
 $+B_{y=0}(G - a) + B_{y=0}(G^{ab} - b)$  if  $a - b$ .

Hence, we have lifted at the multivariate level the recursive definition of Proposition 6 of [ABS04b].

**Remark :** The polynomial  $q$  is not positive : for  $G = a - b$  we have  $q(G) = u'^2 - 2u' + 2v'$ .

## 4.2 Mixing loops and nonloops

We now consider the polynomial  $B_{x=y}(G) := \sigma_{=} (B(G))$  where  $\sigma_{=}$  is the substitution  $[y_a := x_a \text{ for all } a \in V]$ . A direct (static) definition is :

$$B(G) = \sum_{A \cap B = \emptyset} x_{A \cup B} u^{rk(G \nabla B[A \cup B])} v^{n(G \nabla B[A \cup B])}$$

This polynomial has two specializations : first the polynomial  $Q$  of [AvH] defined by  $Q(G, v') = \tau' (B_{x=y}(G))$  where  $\tau'$  is the substitution :

$$[u := 1; v := v' - 2; x_a := y_a := 1 \text{ for all } a \in V]$$

so that :

$$Q(G, v') = \sum_{A \cap B = \emptyset} (v' - 2)^{n(G \nabla B[A \cup B])}.$$

Another one is the *independence polynomial* (Levit and Mandrescu [LM]), expressible by :

$$I(G, v) = \eta(B_{x=y}(G))$$

where  $\eta$  is the substitution  $[u := 0; x_a := 1 \text{ for all } a \in V]$ .

**Proposition 19** : (1)  $B_{x=y}(G \nabla T) = B_{x=y}(G)$  for every graph  $G$  and set of vertices  $T$ .

(2) A graph  $G$  without loops and the polynomial  $B(G)$  can be uniquely determined from  $\rho(B_{x=y}(G))$ , where  $\rho$  replaces  $v$  by 1.

**Proof** : (1) follows from Lemma 4.

(2) Consider two distinct vertices  $a$  and  $b$ . By looking at the ranks of the graphs obtained by adding loops to  $G[\{a, b\}]$ , we see that if  $a - b$ , then we have the monomials  $x_a x_b u$  and  $3x_a x_b u^2$  in  $\rho(B_{x=y}(G))$ . Otherwise, we have the monomials  $x_a x_b$ ,  $2x_a x_b u$  and  $x_a x_b u^2$ .  $\square$

As for Proposition 5, we ask by which algebraic transformation, one can recover  $B(G)$  from  $\rho(B_{x=y}(G))$ .

Corollary 13 yields the following recursive definition :

**Proposition 20** : For every graph  $G$  :

- (1)  $B_{x=y}(\emptyset) = 1$
- (2)  $B_{x=y}(G) = (1 + x_a(u + v))B(G - a)$  if  $N(G, a) = \emptyset$ ,
- (3)  $B_{x=y}(G) = x_a x_b u^2 \{ B_{x=y}(G^{ab} - a - b) + B_{x=y}((G^a)^b - a - b) \}$   
 $+ x_b u \{ B_{x=y}(G^b - b) - B_{x=y}(G^b - a - b) \}$   
 $+ B_{x=y}(G - a) + B_{x=y}(G - b) - B_{x=y}(G - a - b)$  if  $b \in N(G, a)$ .

We wish to compare this definition with that presented in [AvH] for  $Q$  (and for graphs without loops) which we recall :

$$\begin{aligned} \text{(Q1)} \quad & Q(\emptyset) = 1 \\ \text{(Q2)} \quad & Q(G) = u' \cdot Q(G - a) \quad \text{if } N(G, a) = \emptyset, \\ \text{(Q3)} \quad & Q(G) = Q(G - b) + Q(G * b - b) + Q(G^{ab} - a) \quad \text{if } a \in N(G, b). \end{aligned}$$

Proposition 20 yields clearly (Q1) and (Q2) but, instead of (Q3) :

$$\begin{aligned} \text{(Q3')} \quad & Q(G) = Q(G^{ab} - a - b) + Q((G^a)^b - a - b) \\ & + Q(G^b - b) - Q(G^b - a - b) \\ & + Q(G - a) + Q(G - b) - Q(G - a - b) \quad \text{if } b \in N(G, a). \end{aligned}$$

However, Proposition 15 yields for  $G$  with  $a - b$  :

$$\begin{aligned} & Q(G^{ab} - a) = \\ & Q(G - a) - Q(G - a - b) + Q(G^{ab} - a - b) - Q(G^b - a - b) + Q((G^a)^b - a - b), \\ & \text{so that (Q3')} \text{ reduces to} \\ & Q(G) = Q(G - b) + Q(G^b - b) + Q(G^{ab} - a). \end{aligned}$$

It remains to check that :  $Q(G * b - b) = Q(G^b - b)$ . We recall that :

$$\begin{aligned} & G * b = (G \nabla N(G, b))^b = G^b \nabla N(G, b). \text{ Hence :} \\ & Q(G * b - b) = Q(G^b \nabla N(G, b) - b) \\ & = Q((G^b - b) \nabla N(G, b)) \\ & = Q(G^b - b) \end{aligned}$$

by Proposition 19, as was to be proved. Hence, we have established the recursive definition of [AvH], but not at the multivariate level. In order to obtain it from that of Proposition 20, we had to take  $u = 1$  and  $x_a = 1$  for all  $a$ .

The advantage of the definition using (Q1), (Q2), (Q3) is that it only deals with loop-free graphs, whereas the definition of Proposition 20, even if used to compute  $B_{x=y}(G)$  for  $G$  without loops uses the graphs with loops  $(G^a)^b$  and  $G^b$ . It proves also that  $Q$  is positive, which is not obvious from the definition.

### 4.3 The independence polynomial.

The *independence polynomial* is defined by

$$I(G, v) = \sum_k s_k v^k$$

where  $s_k$  is the number of stable sets of cardinality  $k$ . (A looped vertex may belong to a stable set). Hence, we have :

$$I(G, v) = \eta(B_{x=y}(G))$$

where  $\eta$  is the substitution  $[u := 0; x_a := 1 \text{ for all } a \in V]$ .

We let  $B_I(G) = \eta'(B(G))$  where  $\eta'$  is the substitution that replaces  $u$  by 0. It is a multivariate version of the independence polynomial, that can be defined directly by :

$$B_I(G) = \sum_{A, B: \Psi} x_A y_B v^{n(G \nabla B[A \cup B])}$$

where  $\Psi$  is the set of conditions  $A \subseteq V - \text{Loops}(G)$ ,  $B \subseteq \text{Loops}(G)$ ,  $G \nabla B[A \cup B]$  has no edge, so that  $n(G \nabla B[A \cup B]) = |A \cup B|$ . From Corollary 13, we obtain the recursive definition

- (I1)  $B_I(\emptyset) = 1$
- (I2)  $B_I(G) = (1 + x_a v) B_I(G - a)$  if  $N(G, a) = \emptyset$ ,  $a$  is not a loop,
- (I3)  $B_I(G) = (1 + y_a v) B_I(G - a)$  if  $N(G, a) = \emptyset$ ,  $a$  is a loop,
- (I4)  $B_I(G) = B_I(G - a) + B_I(G - b) - B_I(G - a - b)$  if  $b \in N(G, a)$ .

However we can derive alternative reduction formulas :

**Proposition 21** : For every graph  $G$  :

- (I1)  $B_I(\emptyset) = 1$
- (I5)  $B_I(G) = B_I(G - a) + x_a v \cdot B_I(G - a - N(G, a))$ , if  $a$  is not a loop,
- (I6)  $B_I(G) = B_I(G - a) + y_a v \cdot B_I(G - a - N(G, a))$ , if  $a$  is a loop,
- (I7)  $B_I(G) = B_I(G - e) - x_a x_b v^2 \cdot B_I(G - N(G, a) - N(G, b))$ ,  
if  $e$  is the edge  $a - b$  ; (we do not delete  $a$  and  $b$  in  $G - e$ ).

**Proof** : We omit the routine verifications, which use formulas (I1), (I2), (I3) and induction on size of graphs.  $\square$

Formulas (I5), I(6), (I7) are multivariate versions of formulas given in Proposition 2.1 of the survey [LM].

## 5 Computation of interlace and other monadic second-order polynomials

We consider whether and how one can *evaluate* for particular values of indeterminates or *compute* (symbolically) in polynomial time the above defined multivariate interlace polynomials. The results of this section reformulate and precise results from [CMR, Mak04, Mak05, Mak06b]. In particular we consider multivariate polynomials with unbounded numbers of indeterminates.

We define the *size*  $|P|$  of a polynomial  $P$  as the number of its monomials. Since monomials cannot be written in fixed size, this notion of size is a lower bound, not an accurate measure of the size in an actual implementation.

However, it is clear that the multivariate polynomial  $B_{y=0}(G)$  has exponential size in the number of vertices of  $G$ , and so have  $B_{x=y}(G)$  and  $B(G)$ . Hence, we cannot hope for computing them in polynomial time., or by edges in other cases) of the considered graphs), hence as  $|A| + |B|$  in the case of a monomial  $n \cdot x_A y_B u^p v^q$ . For every polynomial  $P(G)$ , we denote by  $P(G) \upharpoonright d$  its  $d$ -truncation defined as the sum of its monomials of quasi-degree at most  $d$ .

For each  $d$ , the polynomials  $B_{y=0}(G) \upharpoonright d$ ,  $B_{x=y}(G) \upharpoonright d$  and  $B(G) \upharpoonright d$  have size less than  $n^{2d}$ , where  $n$  is the number of vertices. Hence, asking for their computations in polynomial time has meaning. Since their monomials have integer coefficients bounded by  $n^{2d}$ , at most  $d$  occurrences of  $G$ -indexed indeterminates  $x_a$ , and indeterminates  $u, v$  with exponents at most  $n$ , we can use their sizes for discussing polynomial time computation of these truncated polynomials.

For their specializations  $P(G)$  where all  $G$ -indexed indeterminates are replaced by constants or ordinary indeterminates  $x, y, u, \dots$ , we have  $P(G) = P(G) \upharpoonright 0$ . Hence, efficient algorithms for computing  $d$ -truncations yield efficient algorithms for computing the classical (non multivariate) versions of these polynomials, and evaluating them for arbitrary values of  $x, y, u, \dots$

**Theorem 22** : For integers  $k, d$ , for each polynomial  $P$  among  $B, B_{y=0}, B_{x=y}, B_I$ , its  $d$ -truncation can be computed in time  $O(|V|^{3d+O(1)})$  for a graph  $G$  of clique-width at most  $k$ . Polynomials  $q(G), Q(G)$  can be computed in times respectively  $O(|V|^7)$  and  $O(|V|^4)$  for graphs of clique-width at most  $k$ .

We will review clique-width in the next section. This theorem gives for each  $d$  a *fixed parameter tractable algorithm* where clique-width (but not  $d$ ) is the parameter. See any of the books [DF] and [FG] for the theory of fixed parameter tractability. As a corollary one obtains the result by Ellis-Monaghan and Sarmiento [ES] that the polynomial  $q$  is computable in polynomial time for *distance-hereditary graphs*, because these graphs have clique-width at most 3, as proved by Golumbic and Rotics [GolRot].

This theorem will be proved by means of an expression of the considered polynomials by formulas of monadic second-order logic and for all multivariate polynomials representable in monadic second-order logic.

## 5.1 Clique-width

*Clique-width* is, like tree-width a graph complexity measure. It is defined and studied in [Cou97, CMR, CouOum, Oum].

Let  $C = \{1, \dots, k\}$  to be used as a set of labels. A  $k$ -graph is a graph  $G$  given with a total mapping from its vertices to  $C$ , denoted by  $lab_G$ . We call  $lab_G(x)$  the *label* of a vertex  $x$ . Every graph is a  $k$ -graph, with all vertices labelled by 1.

For expressing its properties by logical formulas we will handle a  $k$ -graph as a tuple  $(V, A, p_1, \dots, p_k)$  where the adjacency matrix  $A$  is treated as a binary relation ( $A(x, y)$  is true if  $A(x, y) = 1$ , and false if  $A(x, y) = 0$ ) and  $p_1, \dots, p_k$  are unary relations such that  $p_j(x)$  is true iff  $lab_G(x) = j$ .

The operations on  $k$ -graphs are the following ones :

(i) For each  $i \in C$ , we define constants  $\mathbf{i}$  and  $\mathbf{i}^\ell$  for denoting isolated vertices labelled by  $i$ , the second one with a loop.

(ii) For  $i, j \in C$  with  $i \neq j$ , we define a unary function  $add_{i,j}$  such that :

$add_{i,j}(V, A, lab) = (V, A', lab)$  where  $A'(x, y) = 1$  if  $lab(x) = i$  and  $lab(y) = j$  or vice-versa (we want  $A'$  to be symmetric), and  $A'(x, y) = A(x, y)$  otherwise.

This operation adds undirected edges between any two vertices, one labelled by  $i$ , the other by  $j$ , whenever these edges are not already in place.

(iii) We let also  $ren_{i \rightarrow j}$  be the unary function such that

$ren_{i \rightarrow j}(V, A, lab) = (V, A, lab')$  where  $lab'(x) = j$  if  $lab(x) = i$  and  $lab'(x) = lab(x)$  otherwise. This mapping relabels by  $j$  every vertex labelled by  $i$ .

(iv) Finally, we use the binary operation  $\oplus$  that makes the union of disjoint copies of its arguments. (Hence  $G \oplus G \neq G$  and its number of vertices is twice that of  $G$ .)

A well-formed expression  $t$  over these symbols will be called a  $k$ -expression. Its *value* is a  $k$ -graph  $G = val(t)$ . The set of vertices of  $val(t)$  is (or can be defined as) the set of occurrences of the constants (the symbols  $\mathbf{i}$  and  $\mathbf{i}^\ell$ ) in  $t$ . However, we will also consider that an expression  $t$  designates any graph isomorphic to  $val(t)$ . The context specifies whether we consider concrete graphs or graphs up to isomorphism.

The *clique-width* of a graph  $G$ , denoted by  $cwd(G)$ , is the minimal  $k$  such that  $G = val(t)$  for some  $k$ -expression  $t$ . A graph with at least one edge has clique-width at least 2. The graphs  $K_n, S_{n-1}$  ( $= K_{1, n-1}$ ) have clique-width 2, for  $n \geq 3$ . It is clear that clique-width does not depend on loops :  $cwd(G \nabla T) = cwd(G)$  for every set of vertices  $T$ .

The problem of determining if a graph  $G$  has clique-width at most  $k$  is NP-complete if  $k$  is part of the input (Fellows et al. [Fell+]). However, for each  $k$ , there is a cubic algorithm that reports that a graph has clique-width  $> k$  or produces an  $f(k)$ -expression for some fixed function  $f$ . This latter result by Oum [Oum] (improved in [HO]) will fit our purposes.

An *ordered  $k$ -graph*  $G$  is a  $k$ -graph equipped with a linear order  $\leq_G$  on  $V$ . On ordered  $k$ -graphs, we will use the variant  $\overrightarrow{\oplus}$  of  $\oplus$  defined as follows :

(iv)  $G \overrightarrow{\oplus} H$  is the disjoint union of  $G$  and  $H$  with a linear order that extends those of  $G$  and  $H$  and makes the vertices of  $G$  smaller than those of  $H$ .

The other operations are defined in the same way.

This extension will be used as follows (in 5.3 below) : a graph  $G$  being given with a  $k$ -expression  $t$ , we replace everywhere in  $t$  the operation  $\oplus$  by  $\overrightarrow{\oplus}$ . The obtained expression  $\overrightarrow{t}$  defines  $G$  and a linear ordering on its vertices.

## 5.2 Monadic second-order logic

In a few words, *monadic second-order logic* is *first-order logic* over powersets. Formulas are written with special (uppercase) variables denoting subsets of the domains of the considered relational structures, and new atomic formulas of the form  $x \in X$  expressing the membership of  $x$  in a set  $X$  (and for easier reading, also  $x \notin X$ ). For more details see [Cou97] and for a use of logic in a field closely related to that of the present article, see [CouOum].

An ordered  $k$ -graph is handled as a relational structure  $(V, A, \leq, p_1, \dots, p_k)$ . For a  $k$ -graph, we simply omit  $\leq$ . Set variables will thus denote sets of vertices. Here are some examples of graph properties expressed in MS logic.

That  $G$  is *3-vertex colorable* (with neighbour vertices of different colors) can be expressed as  $G \models \gamma$ , read " $\gamma$  is true in the structure  $(V, A)$  representing  $G$ " (here  $\leq, p_1, \dots, p_k$  do not matter) : where  $\gamma$  is the formula :

$$\begin{aligned} & \exists X_1, X_2, X_3 \cdot [\forall x(x \in X_1 \vee x \in X_2 \vee x \in X_3) \wedge \\ & \quad \forall x(\neg(x \in X_1 \wedge x \in X_2) \wedge \neg(x \in X_2 \wedge x \in X_3) \wedge \neg(x \in X_1 \wedge x \in X_3)) \\ & \quad \wedge \forall u, v(A(u, v) \wedge \neg(u = v) \implies \neg(u \in X_1 \wedge v \in X_1) \wedge \neg(u \in X_2 \wedge v \in X_2) \\ & \quad \wedge \neg(u \in X_3 \wedge v \in X_3))]. \end{aligned}$$

That  $G[B]$  (where  $B \subseteq V$ ) is *not connected* can be expressed by the formula  $\delta(X)$ , with  $X$  as free variable :

$$\begin{aligned} & \exists Y \cdot [\exists x \cdot (x \in X \wedge x \in Y) \wedge \exists y(y \in X \wedge y \notin Y) \wedge \\ & \quad \forall x, y \cdot (x \in X \wedge y \in X \wedge A(x, y) \\ & \quad \implies \{(x \in Y \wedge y \in Y) \vee (x \notin Y \wedge y \notin Y)\})]. \end{aligned}$$

For  $B$  a subset of  $V$ ,  $(G, B) \models \delta(X)$ , read " $\delta$  is true in the structure representing  $G$  with  $B$  as value of  $X$ " iff  $G[B]$  is not connected.

For building formulas expressing computations in  $\text{GF}(2)$ , we will also use the set predicate  $\text{Even}(X)$  expressing that the set denoted by  $X$  has even cardinality. This extended language, called *counting modulo 2 monadic second-order logic* is denoted by  $\text{C}_2\text{MS}$ .

**Lemma 23 [CouOum]** : There exists a  $\text{C}_2\text{MS}$  formula  $\rho(X, Y)$  expressing that, in a graph  $G = (V, A)$  we have  $Y \subseteq X$  and the row vectors of  $A[X, X]$  associated with  $Y$  form a basis of the vector space defined by the row vectors of the matrix  $A[X, X]$ . Hence, for each set  $X$ , the sets  $Y$  satisfying  $\rho(X, Y)$  have all the same cardinality, equal to  $\text{rk}(G[X])$ .

**Proof** : We first build a basic formula  $\lambda(Z, X)$  expressing that  $Z \subseteq X$  and the row vectors of  $A[X, X]$  associated with  $Z$  are linearly dependent over  $\text{GF}(2)$ .

Condition  $Z \subseteq X$  is expressed by  $\forall y \cdot (y \in Z \implies y \in X)$ . (We will then use  $\subseteq$  in formulas, although this relation symbol does not belong to the basic syntax).

The second condition is equivalent to the fact that for each  $u \in X$ , the number of vertices  $z \in Z$  such that  $A(z, u) = 1$  is even. This fact is written :

$$\forall u \cdot (u \in X \implies \exists W \cdot [Even(W) \wedge \forall z \cdot (z \in W \iff z \in Z \wedge A(z, u))]).$$

With  $\lambda(Z, X)$  one expresses that  $Y$  (such that  $Y \subseteq X$ ) forms a basis by :

$$\neg \lambda(Y, X) \wedge \forall Z \cdot (\{Y \subseteq Z \wedge Z \subseteq X \wedge \neg(Z \subseteq Y)\} \implies \lambda(Z, X)).$$

We get thus the formula  $\rho(X, Y)$ .  $\square$

We will say that the rank function is definable by a  $C_2MS$ -formula.

All basic results (see Section 5.4) hold for monadic second-order formulas written with set predicates  $Card_p(X)$  expressing that the cardinality of  $X$  is a multiple of  $p$  ( $Even$  is thus  $Card_2$ ). This extension of  $C_2MS$  logic called *counting (modulo) monadic second-order*, can be useful for formalizing computations in fields  $GF(p)$ , along the lines of Lemma 23. For shortness sake we will call *MS formula* a formula of this extended language, (we have no reason to distinguish "pure" monadic second-order formulas from those using counting modulo predicates), and *MS logic* the corresponding language.

A function  $f$  associating a nonnegative integer  $f(A, B, C)$  with every triple of sets  $(A, B, C)$  is *defined by an MS formula*  $\psi(X, Y, Z, U)$  if, for every  $(A, B, C)$  the number  $f(A, B, C)$  is the common cardinality of all sets  $D$  such that  $(G, A, B, C, D) \models \psi(X, Y, Z, U)$ . (We distinguish the variables  $X, Y, Z, U$  from the sets  $A, B, C, D$  they denote). The generalization to functions  $f$  with  $k$  arguments is clear, and the defining formula has then  $k + 1$  free variables.

### 5.3 Multivariate polynomials defined by MS formulas and substitutions

For an MS *formula*  $\varphi$  with free variables among  $X_1, \dots, X_n$ , for a graph  $G$ , we let :

$$\text{sat}(G, \varphi, X_1, \dots, X_n) = \{(A_1, \dots, A_n) \mid A_1, \dots, A_n \subseteq V, \\ (G, A_1, \dots, A_n) \models \varphi(X_1, \dots, X_n)\}.$$

This is the set of all  $n$ -tuples of sets of vertices that satisfy  $\varphi$  in  $G$ . We can write it in the form of a multivariate polynomial :

$$P_\varphi(G) = \sum_{(G, A_1, \dots, A_n) \models \varphi(X_1, \dots, X_n)} x_{A_1}^{(1)} \dots x_{A_n}^{(n)}.$$

It is clear that  $P_\varphi$  describes exactly  $\mathbf{sat}(G, \varphi, X_1, \dots, X_n)$  and nothing else. Its set of indeterminates is  $\mathbf{W}_G$  where  $W = \{x^{(1)}, \dots, x^{(n)}\}$ . The condition describing the summation will be written in a shorter way  $\varphi(A_1, \dots, A_n)$ . Such a polynomial is called a *basic MS polynomial*, and  $n$  is its *order*.

We now show how multivariate polynomials defined by MS formulas can be written as specializations of basic MS-polynomials. To avoid heavy formal definitions, we consider a typical example :

$$P(G) = \sum_{\varphi(A,B,C)} x_A y_B u^{f(A,B,C)} \quad (0)$$

where  $\varphi(X, Y, Z)$  is an MS formula and  $f$  is a function on triples of sets defined by an MS formula  $\psi(X, Y, Z, U)$ . After usual summation of *similar* monomials (those with same indeterminates with same exponents) the general monomial of  $P(G)$  is of the form  $c \cdot x_A y_B u^p$  where  $c$  is the number of sets  $C$  such that  $f(A, B, C) = p$ . We first observe that  $P(G) = \sigma(P'(G))$  where :

$$P'(G) = \sum_{\varphi(A,B,C)} x_A y_B z_C u^{f(A,B,C)}$$

where  $\sigma$  replaces each  $z_c$  by 1. We are looking for an expression of  $P(G)$  as  $\mu(\sigma(P_\theta(G))) = \mu \circ \sigma(P_\theta(G))$  where  $\mu$  replaces each  $u_d$  by  $u$  in :

$$P_\theta(G) = \sum_{\theta(A,B,C,D)} x_A y_B z_C u_D$$

for some formula  $\theta(X, Y, Z, U)$ . Taking  $\theta(X, Y, Z, U)$  to be  $\varphi(X, Y, Z) \wedge \psi(X, Y, Z, U)$  would be incorrect in cases where several sets  $D$  satisfy  $\psi(A, B, C, D)$  for a triple  $(A, B, C)$  satisfying  $\varphi$ . We overcome this difficulty in the following way : we let  $V$  be linearly ordered in an arbitrary way ; we let  $\psi'$  be the formula, written with a new binary relation symbol denoting the order on  $V$  such that  $\psi'(X, Y, Z, U)$  is equivalent to :

$$\psi(X, Y, Z, U) \wedge \forall T \cdot [\psi(X, Y, Z, T) \implies "U \leq_{lex} T"]$$

where  $U \leq_{lex} T$  means that  $U$  is less than or equal to  $T$  in the lexicographic order derived from the ordering of  $V$ . This is easily expressed by an MS formula. The formula  $\psi'(X, Y, Z, U)$  defines the function  $f$  by selecting a unique set  $D$  such that  $f(A, B, C) = |D|$ . This set is unique for each linear order on  $V$  by the hypothesis on  $\psi$ , but its cardinality does not depend on the chosen order. Hence we have the desired expression of  $P$  :

$$P(G) = \mu \circ \sigma \left( \sum_{\varphi(A,B,C) \wedge \psi'(A,B,C,D)} x_A y_B z_C u_D \right).$$

These remarks motivate the following definition :

**Definition 23 a :** *MS-polynomials*

A *multivariate MS-polynomial* is a polynomial of the form :

$$P(G) = \sum_{\varphi(A_1, \dots, A_m)} x_{A_1}^{(1)} \dots x_{A_{m'}}^{(m')} u_1^{f_1(A_1, \dots, A_m)} \dots u_p^{f_p(A_1, \dots, A_m)} \quad (1)$$

where  $\varphi(X_1, \dots, X_m)$  is an MS-formula,  $m' \leq m$  and  $f_1, \dots, f_p$  are MS definable functions. A *multivariate MS-polynomial in normal form* is defined as  $P = \sigma \circ P_\varphi$  where :

$$P_\varphi(G) := \sum_{\varphi(A_1, \dots, A_m)} x_{A_1}^{(1)} \dots x_{A_m}^{(m)} \quad (2)$$

for  $\varphi$  an MS-formula and  $\sigma$  a substitution that can replace a variable  $x_a$  by 1 or by an ordinary variable say  $u$ . From the above observations, it is clear that every MS-polynomial can be written in normal form, for graphs arbitrarily ordered. (The expression of  $P$  is said to be *order-invariant*.)

A *generalized multivariate MS-polynomial* is a polynomial defined as  $P = \sigma \circ P_\varphi$  where  $P_\varphi$  is as in (2) for  $\varphi$  an MS-formula and  $\sigma$  is a specializing substitution. A generalized MS-polynomial may have negative coefficients, and in this case, cannot have a normal form.

**Question 23 b :** Is it true that every positive generalized MS-polynomial (i.e., that is positive for every graph) has an expression in normal form ?

We say that  $P$  is of *order*  $m$  if it can be expressed as  $P = \sigma \circ P_\varphi$  where  $\varphi$  has  $m$  free variables. Hence a polynomial of the form (1) above is of order  $m + p$ .

Then for a graph  $G$  with  $n$  vertices and  $P$  defined by (1),  $P(G)$  has size at most  $2^{m'n}$ , degree at most  $n(m' + p)$ , positive coefficients of value at most  $2^{mn}$ . These bounds will be useful for evaluating the cost of computations of truncations of polynomials.

#### *Transformations of MS-polynomials.*

We now show how some specializations can be reflected by transformations of the defining formulas. We review some cases which arised in the present article, by taking a polynomial  $P$  of the form (0). For each case 1, ..., 4 we denote by  $P_i$  the polynomial obtained from  $P$ .

*Case 1 :*  $x_a := 0$ . We have :

$$P_1(G) = \sum_{\varphi'(B, C)} y_B u^{f(\varnothing, B, C)}$$

where  $\varphi'(Y, Z)$  is defined as  $\exists X \cdot [\varphi(X, Y, Z) \wedge \forall z \cdot z \notin X]$  so that  $\varphi'(B, C)$  is equivalent to  $\varphi(\varnothing, B, C)$ . The function  $f(\varnothing, Y, Z)$  is defined by  $\exists X \cdot [\psi(X, Y, Z, U) \wedge \forall z \cdot z \notin X]$ .

*Case 2 :*  $u := 0$ . We have

$$P_2(G) = \sum_{\varphi'(A, B, C)} x_A y_B$$

where  $\varphi'(X, Y, Z)$  is  $\varphi(X, Y, Z) \wedge \exists U \cdot [\psi(X, Y, Z, U) \wedge \forall z \cdot z \notin U]$ , which is equivalent to  $\varphi(X, Y, Z) \wedge f(X, Y, Z) = 0$ .

Case 3 :  $x_a := 1$ . We have

$$P_3(G) = \sum_{\varphi(A,B,C)} y_B u^{f(A,B,C)}.$$

The sets  $A$  are, like the sets  $C$ , "invisible" in the monomials. However, they play a role. They contribute to the multiplicity of the monomials  $y_B u^p$ , for  $p = f(A, B, C)$ . (This case has been considered above at the beginning of Section 5.3).

Case 4 :  $x_a := y_a$ . Here

$$P_4(G) = \sum_{\varphi(A,B,C)} x_{A \cup B} u^{f(A,B,C)}$$

We assume here, and this is enough for this article, that  $\varphi(A, B, C)$  implies that  $A$  and  $B$  are disjoint. Then to reach the general syntax we write  $P_4(G)$  as follows

$$P_4(G) = \sum_{\varphi'(D,B,C)} x_D u^{g(D,B,C)}$$

where  $D$  stands for  $A \cup B$ ,  $\varphi'(W, Y, Z)$  is chosen to be equivalent to the formula  $Y \subseteq W \wedge \varphi(W - Y, Y, Z)$  and the function  $g$  is defined by :

$$g(D, B, C) = f(D - B, B, C)$$

whence also by a formula  $\psi'(W, Y, Z, U)$  equivalent to :

$$\exists X \cdot [\psi(X, Y, Z, U) \wedge X \subseteq W \wedge Y = W - X].$$

**Lemma 24** : Each polynomial  $P$  among  $B$ ,  $B_{y=0}$ ,  $B_{x=y}$ ,  $B_I$  is an MS-polynomial. For every graph  $G$ , ordered in an arbitrary way, we have an expression of  $P$  in normal form  $P(G) = \sigma \circ P_\theta(G)$  for an MS formula  $\theta$  expressing properties of ordered graphs, and some specializing substitution  $\sigma$ .

Note that  $P(G) = \sigma(P_\theta(G))$  for every linear order on  $G$ .

**Proof** : We only consider  $B$ . The other cases follow by the techniques presented above. We recall the definition of  $B$ :

$$B(G) = \sum_{A \cap B = \emptyset} x_A y_B u^{rk(G \nabla B[A \cup B])} v^{n(G \nabla B[A \cup B])}$$

We let

$$P_\theta(G) = \sum_{\theta(A,B,C,D)} x_A y_B u^C v^D$$

where  $\theta(A, B, C, D)$  holds iff  $A \cap B = \emptyset$ ,  $C \subseteq A \cup B$ ,  $D = A \cup B - C$ ,  $C$  is the smallest basis of the vector space spanned by  $M_{G \nabla B}[A \cup B, A \cup B]$ , where  $M_{G \nabla B}$  is the adjacency matrix of  $G \nabla B$ . By "smallest" we mean with respect to the lexicographic ordering derived from the ordering of  $G$ . It follows that  $|C| = rk(G \nabla B[A \cup B])$  and  $|D| = n(G \nabla B[A \cup B])$ . By Lemma 23, one can express these conditions by an MS formula  $\theta$ .

Hence,  $B = \sigma \circ P_\theta$  where  $\sigma$  replace  $u_a$  by  $u$  and  $v_a$  by  $v$ .  $\square$

## 5.4 The Fefermann and Vaught paradigm applied to MS-polynomials

We now show how multivariate polynomials defined by MS formulas can be computed "efficiently", not by reduction formulas, because they yield (when they exist) exponential computations in general, but by induction on  $k$ -expressions defining the considered graphs. This will only apply to graphs of clique-width bounded by a fixed value. We will use the *Fefermann-Vaught paradigm*, presented in detail by Makowsky [Mak04, Mak05].

We need operations that manipulate sets of  $q$ -tuples, in particular, those of the form  $\mathbf{sat}(G, \varphi, X_1, \dots, X_q)$ , and, equivalently as we will see, the polynomials  $P_\varphi(G)$ .

For sets  $R, S$  and  $S' \subseteq \mathcal{P}(V)^q$ , we write

$R = S \uplus S'$  if  $R = S \cup S'$  and  $S \cap S' = \emptyset$ , and

$R = S \boxtimes S'$  if  $S \subseteq \mathcal{P}(V_1)^q$ ,  $S' \subseteq \mathcal{P}(V_2)^q$ ,  $V_1 \cap V_2 = \emptyset$ , and  $R$  is the set of  $q$ -tuples  $(A_1 \cup B_1, \dots, A_q \cup B_q)$  such that  $(A_1, \dots, A_q) \in S$  and  $(B_1, \dots, B_q) \in S'$ .

For each  $S \subseteq \mathcal{P}(V)^q$ , we let  $P(S)$  be the multivariate polynomial :

$$P(S) = \sum_{(A_1, \dots, A_q) \in S} x_{A_1}^{(1)} \dots x_{A_q}^{(q)} .$$

Clearly,  $P_\varphi(G) = P(\mathbf{sat}(G, \varphi, X_1, \dots, X_q))$ . The following is clear :

**Fact 25** : For  $S, S' \subseteq \mathcal{P}(V)^q$ , we have  $P(S \uplus S') = P(S) + P(S')$  and  $P(S \boxtimes S') = P(S) \cdot P(S')$ .

We denote by  $\mathbf{U}_k$  the (finite) set of unary operations allowed in  $k$ -expressions. We denote by  $MS(k, q)$  the set of MS formulas written with the basic symbols  $=, \in, \notin, Card_p$  and the relation symbols  $A, <, p_1, \dots, p_k$  (hence able to express properties of ordered  $k$ -graphs) with free variables in the set  $\{X_1, \dots, X_q\}$ . The following theorem discussed in [Mak04] is proved in [Cou97, CouMos] in closely related forms :

**Theorem 26** : For every  $k, q$ , for every formula  $\xi$  in  $MS(k, q)$ , there exists a finite subset  $\Phi$  of  $MS(k, q)$  containing  $\xi$  and satisfying the following properties:

(1) For every  $\varphi \in \Phi$  for every  $op \in \mathbf{U}_k$  there exists a formula  $\varphi^{op} \in \Phi$  such that, for every ordered  $k$ -graph  $G$  :

$$\mathbf{sat}(op(G), \varphi, X_1, \dots, X_q) = \mathbf{sat}(G, \varphi^{op}, X_1, \dots, X_q).$$

(2) For every  $\varphi \in \Phi$  there exist  $p$  and  $(\theta_1, \dots, \theta_p, \psi_1, \dots, \psi_p) \in \Phi^{2p}$  such that for disjoint ordered  $k$ -graphs  $G$  and  $H$  :

$$\mathbf{sat}(G \overset{\rightarrow}{\oplus} H, \varphi, X_1, \dots, X_q) = \uplus_{1 \leq i \leq p} \mathbf{sat}(G, \theta_i, X_1, \dots, X_q) \boxtimes \mathbf{sat}(H, \psi_i, X_1, \dots, X_q).$$

These statements also hold for (unordered)  $k$ -graphs and the operation  $\oplus$  instead of  $\overset{\rightarrow}{\oplus}$ .

Let  $\Phi$  be a set of formulas as in Theorem 26. We get a finite family of polynomials  $(P_\varphi)_{\varphi \in \Phi}$  that satisfy mutually recursive computations rules. Actually, the recursive rules apply to the family of polynomials  $(\sigma \circ P_\varphi)_{\varphi \in \Phi}$  where  $\sigma$  is a specializing substitution. We recall that by a polynomial we mean (ambiguously) a mapping  $P$  associating with a graph  $G$  a polynomial in  $\mathbf{Z}[U \cup \mathbf{W}_G]$ .

**Corollary 27** : Let  $\Phi$  satisfy the properties of Theorem 26. Let  $\sigma$  be a specializing substitution. We have the following computation rules :

(1) For every  $\varphi \in \Phi$ , for every  $op \in \mathbf{U}_k$  for every ordered  $k$ -graph  $G$  :

$$(\sigma \circ P_\varphi)(op(G)) = (\sigma \circ P_{\varphi \circ op})(G).$$

(2) For every  $\varphi \in \Phi$  for every disjoint ordered  $k$ -graphs  $G$  and  $H$  :

$$(\sigma \circ P_\varphi)(G \overrightarrow{\oplus} H) = \sum_{1 \leq i \leq p} (\sigma \circ P_{\theta_i})(G) \cdot (\sigma \circ P_{\psi_i})(H).$$

where  $\varphi^{op}$  and  $(\theta_1, \dots, \theta_p, \psi_1, \dots, \psi_p)$  are as in Theorem 26.

**Proof** : If  $\sigma$  is the identity substitution, then (1) and (2) are direct translations of (1) and (2) of Theorem 26.

Since  $\sigma \circ (P+Q) = \sigma \circ P + \sigma \circ Q$  i.e.,  $(\sigma \circ P + \sigma \circ Q)(G) = \sigma(P(G)) + \sigma(Q(G)) = \sigma((P+Q)(G))$  and similarly,  $\sigma \circ (P \cdot Q) = (\sigma \circ P) \cdot (\sigma \circ Q)$ , for every substitution  $\sigma$  and every two polynomials  $P, Q$ , the equalities extend to the general case as stated.  $\square$

Hence, this corollary concerns all multivariate polynomials described in 5.3. We will use it for their computation.

## 5.5 Computing polynomials in polynomial time

We discuss the computation of polynomials written in the form  $\sigma \circ P_\xi$  for  $P_\xi$  a basic MS-polynomial of order  $q$ , and a substitution  $\sigma$ . This will also apply to *evaluations* of polynomials where all indeterminates are given some numeric value, either integer, real or complex.

The number  $n$  will denote the number of vertices of the considered graph  $G$ . In evaluating the cost of a computation as  $O(e)$  where  $e$  is a nonnegative expression, we omit the case where  $e$  might have value 0 (rigourously, we should write  $O(e+1)$ ). Similarly  $\log(e)$  stands for  $\text{Max}\{1, \log(|e|+1)\}$ .

Let a graph  $G$  be given by a  $k$ -expression  $t$ . We "order"  $t$  into  $\overrightarrow{t}$  which defines  $G$  with a linear order of its vertices (cf. Section 5.1).

It is clear that for each constant,  $\mathbf{i}$  or  $\mathbf{i}^\ell$ , each polynomial  $(\sigma \circ P_\varphi)(\mathbf{i}^\ell)$  can be computed from the definitions. We can thus compute for each subterm  $s$  of  $\overrightarrow{t}$  the family of polynomials  $((\sigma \circ P_\varphi)(\text{val}(s)))_{\varphi \in \Phi}$ . In particular, at the end of the computation, one gets  $(\sigma \circ P_\varphi)(G)$  for all  $\varphi \in \Phi$ . (Which is too much. A method for restricting similar computations to their necessary parts is described in [CouMos] or in [Mak04], Definition 4.17.)

This computation uses at most  $n \cdot |\Phi|$  times the computation rules of Corollary 27(2), because in a term, the number of occurrence of  $\overrightarrow{\oplus}$  is  $s-1$ ,

where  $s$  is the number of occurrences of constants, which is equal to  $|V| = n$ . Here,  $|\Phi|$  is constant. The computation time is bounded by  $2n \cdot c_G \cdot p_{max} \cdot |\Phi|$ , where  $p_{max}$  is the maximum value of  $p$  in the rules of Theorem 26(2), and  $c_G$  bounds the cost of the addition and of the multiplication of two polynomials. This bound depends on  $G$ . When do we obtain a polynomial algorithm in  $n$ ?

We first precise the way we will count the cost of operations on polynomials. Using *unit cost measure*, we will count for one the cost of each basic operation on numbers : comparison, addition, subtraction, multiplication. The cost of evaluating  $x^m$  for a positive integer  $m$  is thus  $O(\log(m))$ . In the computation of a truncated polynomial  $P \upharpoonright d$ , we will consider  $d$  as "relatively small" and fixed, like  $|\Phi|$  (and actually much smaller in potential applications.) Hence we will count for one the cost of computing  $x^m$  for  $m \leq d$ .

However, for computing a polynomial or evaluating it for integer values of the arguments, we can also use the *real cost measure* and consider that the cost of a comparison, an addition or a subtraction of two positive integers  $x$  and  $y$  is  $O(\log(|x| + |y|))$  and that of their multiplication is  $O(\log(xy))$ . Coefficients of polynomials may be exponentials in the sizes of the considered graphs. However, in situations where the absolute values of coefficients and exponents are no larger than  $2^{p(n)}$  for fixed polynomials  $p$ , a polynomial bound on computation time with respect to unit cost measure remains polynomial with respect to real cost measure. The exponents of the polynomial bounds are just larger.

We will base the following estimations of the cost of computations on straightforward data structures : a polynomial is a uniquely defined list of monomials sorted by increasing order of quasi-degree, where two monomials of same quasi-degree are ordered lexicographically. Each monomial is written in a canonical way by means of a fixed ordering of indeterminates. We deal with monomials with a variable number of indeterminates, however, this number is always bounded by  $n \cdot |X| + |U|$  and where  $\mathbf{X}_G$  is the set of  $G$ -indexed indeterminates, and  $U$  the set of others.

The basic operations on pairs of monomials  $m, m'$  are comparison, summation of coefficients if  $m, m'$  are *similar* monomials (if they have same indeterminates with same respective degrees), and multiplication. For a monomial  $m$ , we denote by  $\mathbf{v}(m)$  the number of its indeterminates. The costs of these operations are respectively  $O(\mathbf{v}(m) + \mathbf{v}(m'))$ ,  $1$ ,  $O(\mathbf{v}(m) + \mathbf{v}(m'))$ . We denote by  $\mathbf{v}(P)$  the number of indeterminates in a polynomial  $P$ .

- Lemma 28** : (1) For every  $P, Q, d$ , if  $\mathbf{v}(P), \mathbf{v}(Q) \leq \mathbf{v}_{max}$  we have:  
 $(P + Q) \upharpoonright d = P \upharpoonright d + Q \upharpoonright d$  and  $(PQ) \upharpoonright d = ((P \upharpoonright d) \cdot (Q \upharpoonright d)) \upharpoonright d$ .  
(2) Computing  $P + Q$  takes time  $O(\mathbf{v}_{max} \cdot (|P| + |Q|))$ .  
(3) Computing  $PQ$  takes time :  $O(\mathbf{v}_{max} \cdot |P|^2 \cdot |Q|)$  if  $|P| \leq |Q|$ .

**Proof** : (1) Clear from the definitions.

(2) Note that  $|P + Q| \leq |P| + |Q|$ . The addition of  $P$  and  $Q$  is done by interleaving their lists of monomials and by adding the coefficients of similar monomials. This gives the result by the remarks on the cost of operations on monomials.

(3) Let  $|P| \leq |Q|$  and  $\mathbf{v}(P), \mathbf{v}(Q) \leq \mathbf{v}_{\max}$ . Note that  $|PQ| \leq |P| \cdot |Q|$ . We compute  $PQ$  by multiplying  $|P|$  times the polynomial  $Q$  by a monomial of  $P$ , and by performing  $|P| - 1$  additions of polynomials of size at most  $|P| \cdot |Q|$ . The time taken is at most

$$O(\mathbf{v}_{\max} \cdot |P| \cdot |Q| + (|P| - 1) \cdot \mathbf{v}_{\max} \cdot |P| \cdot |Q|) = O(\mathbf{v}_{\max} \cdot |P|^2 \cdot |Q|). \quad \square$$

**Remark :** If in (3)  $P$  and  $Q$  are positive, one gets the bound  $O(\mathbf{v}_{\max} \cdot |P| \cdot |PQ|)$  because all intermediate results have size at most  $|PQ|$ .

**Theorem 29 :** Let  $k, d$  be fixed integers. The  $d$ -truncation of a generalized multivariate MS-polynomial  $P(G)$  can be computed in time  $O(n^{6d+O(1)})$  for every graph  $G$  of clique-width at most  $k$ .

The constants hidden by the  $O$ -notation depend on  $P$  and  $k$ . The particular case of evaluations (for numerical values of indeterminates) will be discussed later. Closely related formulations of this theorem are in [CMR, Mak04, Mak05]. For a polynomial  $P$ , we define  $\|P\| := (|P|, \deg(P), C_{\max}(P))$  where  $\deg(P)$  is the degree of  $P$  and  $C_{\max}(P)$  the maximum absolute value of its coefficients. Triples of integers are ordered componentwise.

**Lemma 30 :** Let  $P$  be a polynomial and  $\sigma$  be a substitution such that  $\|\sigma(x)\| \leq (\mathbf{s}_{max}, \mathbf{d}_{max}, \mathbf{c}_{max})$  for every indeterminate  $x$ . The polynomial  $\sigma \circ P$  satisfies :

$$\|\sigma \circ P\| \leq (|P| \cdot (\mathbf{s}_{max})^{\deg(P)}, \deg(P) \cdot \mathbf{d}_{max}, |P| \cdot C_{\max}(P) \cdot (\mathbf{s}_{max} \mathbf{c}_{max})^{\deg(P)}).$$

**Proof :** Easy verification.  $\square$

It follows from previous observations (see the remark after Question 23 b) that if  $P_\varphi$  has order  $m$ , then for every graph  $G$ , the polynomial  $\sigma(P_\varphi(G))$  has size and coefficients bounded by  $2^{O(n)}$  and degree bounded by  $n(m' + p) \cdot \mathbf{d}_{max} = O(n)$  ( $m' \leq m$  and  $p$  are as in that remark). We can thus use the unit cost measure.

**Proof of Theorem 29 :** Let  $k, d$  be integers, let  $P$  be a polynomial expressed as  $\sigma \circ P_\xi$  for an MS formula  $\xi$  and a substitution  $\sigma$ . We aim at computing its  $d$ -truncation. Let  $\Phi$  be the corresponding set of formulas as in Theorem 26. We observe that by Lemma 28 (1), Corollary 27 yields :

- (1) For every  $\varphi \in \Phi$ , for every  $op \in \mathbf{U}_k$  for every ordered  $k$ -graph  $G$  :  
 $\sigma(P_\varphi(op(G))) \upharpoonright d = \sigma(P_{\varphi \circ p}(G)) \upharpoonright d$ .
- (2) For every  $\varphi \in \Phi$  for every disjoint ordered  $k$ -graphs  $G$  and  $H$  :  
 $\sigma(P_\varphi(G \overline{\oplus} H)) \upharpoonright d = \sum_{1 \leq i \leq p} ((\sigma(P_{\theta_i}(G)) \upharpoonright d) \cdot (\sigma(P_{\psi_i}(H)) \upharpoonright d)) \upharpoonright d$ .

Note that we do not have :  $\sigma(P(G)) \upharpoonright d = \sigma(P(G) \upharpoonright d)$  in general. As observed above, the time to compute  $\sigma(P(G))$  is  $2n \cdot c_G \cdot p_{max} \cdot |\Phi|$ , where  $c_G$  bounds the costs of adding and multiplying the polynomials occurring in recursion rules (1) and (2). We need only count multiplications which are more costly than additions and in proportional number. Let us assume that all the polynomials  $\sigma(P_\varphi(G))$  are in  $\mathbf{Z}[\mathbf{X}_G \cup U]$ . By Lemma 28 we have, for the  $d$ -truncations of all such polynomials  $P, Q$  :

- (i)  $c_G = O(\mathbf{v}_{max} \cdot |P \upharpoonright d|^2 \cdot |Q \upharpoonright d|)$  with
- (ii)  $\mathbf{v}_{max} = n \cdot |X| + |U| = O(n)$ , and
- (iii)  $|P \upharpoonright d| = O(n^{2d+|U|})$ .

For proving (iii), we note that a monomial of  $P \upharpoonright d$  is a product of at most  $d$  factors of the form  $x_a^s$  and of at most  $|U|$  factors of the form  $u^s$ , in both cases for  $s \leq \deg(P) \cdot \mathbf{d}_{max}$ . There are  $n \cdot |X| \cdot \deg(P) \cdot \mathbf{d}_{max}$  factors of the form  $x_a^s$  and  $\deg(P) \cdot \mathbf{d}_{max}$  factors  $u^s$  for each  $u$ . Hence  $|P \upharpoonright d| = O((n \cdot \deg(P))^d \cdot \deg(P)^{|U|}) = O(n^{2d+|U|})$  since  $\deg(P) = O(n)$ .

This gives for  $2n \cdot c_G \cdot p_{max} \cdot |\Phi|$  the bound  $O(n^{2+6d+3|U|})$ .

We must take into account the cost of building for a graph  $G$  of clique-width at most  $k$  a clique-width expression. In cubic time, one can construct for graphs of clique-width at most  $k$  an  $f(k)$ -expression, for a fixed function  $f$  ([HO, Oum]). This suffices for our purposes. The total time is thus  $O(n^t)$  where  $t = \text{Max}\{3, 2 + 6d + 3 \cdot |U|\}$ . (This bound applies if  $d = |U| = 0$ .)  $\square$

We extend this result to numerical evaluations. Let  $P(G)$  be a polynomial in  $\mathbf{Z}[\mathbf{X}_G \cup U]$  to be evaluated for a graph  $G$ . An *evaluating substitution*  $\nu$  replaces indeterminates by integer, real or complex values. Let  $\nu$  be such a mapping. One can consider  $\nu(P(G))$  as a polynomial reduced to a constant, that is the desired value of  $P(G)$  for the values of indeterminates specified by  $\nu$ . Note that  $\nu(P(H))$  is well-defined for every graph  $H$  with set of vertices included in the set  $V$  of vertices of  $G$ . This remark will be useful for the computation of  $\nu(P(G))$  by induction on the structure of  $G$  using Corollary 27. The costs of computations are the same for polynomials with integer, real or complex values since we use unit cost measure.

**Corollary 31** : Let  $k$  be an integer. For every generalized multivariate MS-polynomial  $P$  and every evaluating substitution, the corresponding value of  $P(G)$  for a graph  $G$  of clique-width at most  $k$  can be computed in cubic time in the number of vertices of  $G$ . It can be computed in linear time if the graph is given by a  $k$ -expression.

**Proof** : Let  $\nu$  be an evaluating substitution. It associates a number with each  $u$  in  $U$  and each  $x_a$  in  $\mathbf{X}_G$ , for a given graph  $G$ . As in the proof of Theorem 29 we have :

(1) For every  $\varphi \in \Phi$ , for every  $op \in \mathbf{U}_k$  for every ordered  $k$ -graph  $H$  with  $V_H \subseteq V_G$  :

$$\nu(P_\varphi(op(H))) = \nu(P_{\varphi \circ p}(H)).$$

(2) For every  $\varphi \in \Phi$  and all disjoint ordered  $k$ -graphs  $H$  and  $H'$  with  $V_H, V_{H'} \subseteq V_G$  :

$$\nu(P_\varphi(\overrightarrow{H \oplus H'})) = \sum_{1 \leq i \leq p} \nu(P_{\theta_i}(H)) \cdot \nu(P_{\psi_i}(H')).$$

Here we compute values, not polynomials. The cost is thus  $2n \cdot p_{max} \cdot |\Phi| = O(n)$  assuming known a  $k$ -expression defining  $G$ .  $\square$

We now precise the bounds for the polynomial  $B$  and its specializations. Theorem 22 is actually a corollary of Theorem 29.

**Proof of Theorem 22 :**

For every graph  $G$  with  $n$  vertices, we have the following bounds :

$$\begin{aligned} \|B(G)\| &\leq (3^n, 2n, 1), |B(G) \upharpoonright d| = O(n^{d+2}), \\ \|B_{y=0}(G)\| &\leq (2^n, 2n, 1), |B_{y=0}(G) \upharpoonright d| = O(n^{d+1}), \\ \|B_{x=y}(G)\| &\leq (n2^n, 2n, 2^n), |B_{x=y}(G) \upharpoonright d| = O(n^{d+2}), \\ \|B_I(G)\| &\leq (2^n, 2n, 1), |B_I(G) \upharpoonright d| = O(n^{d+1}), \\ \|q(G)\| &\leq (n^2, 2n, 2^n), \\ \|Q(G)\| &\leq (n+1, n, 3^n). \end{aligned}$$

As in the proof of Theorem 29, we need only bound the costs  $c_G$  of the multiplications of polynomials. From these evaluations, we get the bounds  $O(n^{3d+8})$  for  $B$  and  $B_{x=y}$ ,  $O(n^{3d+5})$ , for  $B_{y=0}$  and  $B_I$ ,  $O(n^7)$  for  $q$  and  $O(n^4)$  for  $Q$  for the last two,  $\mathbf{v}_{max}$  is constant.  $\square$

*Remark about the size of constants.*

Sets  $\Phi$  in Theorem 26 are very large if they are constructed in a blind manner from an MS formula : the size is a tower of exponentials proportional to the quantification depth of formula  $\xi$ . However, if alternatively a family of polynomials  $\sigma \circ P_\varphi$  satisfying Corollary 27 is constructed directly, by using our knowledge of the meaning of the properties defined by formula  $\xi$ , then one may obtain a usable recursive definition. Hence the above estimations leave a great space for improvements.

## 6 Conclusion

We have defined a multivariate interlace polynomial that generalizes the existing interlace polynomials. The multivariate methodology puts in light the *meaning of polynomials*. Classical polynomials are degraded versions of multivariate ones. The multivariate approach is well-adapted to the logical description of polynomials. And the use of monadic second-order logic yields FPT algorithms for evaluating polynomials at particular values or for computing significant portions of them, called truncations.

For computing such polynomials in full, one might use linear delay enumeration algorithms and try to obtain monomials one by one, by increasing degree, with a delay between two outputs linear in the size of the next output. Such algorithms are considered in [Cou06] and [Bag] for MS definable problems and graphs of bounded clique-width.

Another research perspective consists in enriching the notion of configuration. In this article, in particular in Section 5, a configuration is an  $m$ -tuple of subsets for fixed  $m$ . One could try to extend the methodology of Section 5.3 to more complex configurations like partitions of unbounded size or permutations.

Finally, in order to build a zoology as opposed to maintaining a zoo (as written by J. Makowsky [Mak06a]) it is important to relate the various polynomials by means of algebraic reductions (specializations), or logical reductions or transformations of other kind.

**Acknowledgements** : This work is part of a larger project concerning the logical definition and the complexity of computation of graph polynomials conducted with J. Makowsky. I thank S. Oum for the equalities of Lemma 2(4). Thanks to M. Las Vergnas for useful comments.

## 7 References

[ABS] R Arratia, B Bollobás, G. Sorkin : The interlace polynomial: a new graph polynomial. *J. of Comb. Theory B* **92** (2004) 199-233.

[ABS04b] R Arratia, B Bollobás, G. Sorkin : A Two-Variable Interlace Polynomial, *Combinatorica* **24** (2004) 567-584

[And] A. Andrzejak, An algorithm for the Tutte polynomials of graphs of bounded treewidth, *Discrete Mathematics*, **190** (1998) 39-54.

[AvH] M. Aigner, H. van der Holst, Interlace polynomials, *Linear algebra and applications* **377** (2004) 11-30.

[Bag] G. Bagan, MSO queries on tree decomposable structures are computable with linear delay, *Proceedings of Computer Science Logic 2006*, *Lec. Notes Comput. Sci.* **4207** (2006) 167-181.

[Bjo] A. Björner, The homology and shellability of matroids and geometric lattices, Chapter 7 of [Whi].

[Bou] A. Bouchet, Circle graph obstructions, *Journal of Combinatorial Theory Series B* **60** (1994) 107-144.

[Cou97] B. Courcelle, The expression of graph properties and graph transformations in monadic second-order logic, Chapter 5 of the "Handbook of graph grammars and computing by graph transformations, Vol. 1 : Foundations", G. Rozenberg ed., World Scientific (New-Jersey, London), 1997, pp. 313-400.

[Cou06] B. Courcelle, Linear delay enumeration and monadic second-order logic, March 2006, to appear in Discrete Applied Mathematics.

[CMR] B. Courcelle, J. A. Makowsky, Udi Rotics: On the fixed parameter complexity of graph enumeration problems definable in monadic second-order logic. Discrete Applied Mathematics **108** (2001) 23-52

[CGM] : B. Courcelle, B. Godlin, J. A. Makowsky : In preparation.

[CouMos] B. Courcelle, M. Mosbah: Monadic Second-Order Evaluations on Tree-Decomposable Graphs. Theor. Comput. Sci. **109** (1993) 49-82.

[CouOum] B. Courcelle, S. Oum, Vertex-minors, monadic second-order logic and a conjecture by Seese, J. comb. Theory B **97** (2007) 91-126.

[DF] R. Downey et M. Fellows, *Parameterized Complexity*, Springer-Verlag, 1999

[ES] J. Ellis-Monaghan, I. Sarmiento, Distance Hereditary Graphs and the Interlace Polynomial, Preprint, arXiv:math.CO/0604088 v1 4 Apr 2006.

[Fell+] M. Fellows, F. Rosamond, U. Rotics, S. Szeider Clique-width minimization is NP-hard, Proceedings of the thirty-eighth annual ACM symposium on Theory of computing, Seattle, 2006, pp. 354 - 362.

[FG] J. Flum, M. Grohe, *Parameterized complexity theory*, Springer, 2006.

[GHN] O. Gimenez, P. Hliněný, M. Noy, Computing the Tutte polynomial on graphs of bounded clique-width, Proceedings WG 2005, Lec. Notes Comput. Sci. **3787** (2005) 59-68.

[GolRot] M. Golumbic, U. Rotics: On the Clique-Width of Some Perfect Graph Classes. Int. J. Found. Comput. Sci. **11** (2000): 423-443

[Hli03] P. Hliněný, On Matroid Properties Definable in the MSO Logic. Proceedings MFCS 2003, Lec. Notes Comp. Sci. **2747** (2003) 470-479

[Hle06] P. Hliněný, The Tutte polynomial for matroids of bounded branch-width, Combin. Prob. Computing **15** (2006) 397-409 (Cambridge Univ. Press).

[HO] P. Hliněný, S. Oum, Finding branch-decompositions and rank-decompositions, Preprint, March 2007.

[LV] M. Las Vergnas, Le polynôme de Martin d'un graphe eulérien, *Annals of Discrete Mathematics* **17** (1983), 397-411.

[LM] V. E. Levit, E. Mandrescu: The independence polynomials : a survey, Preprint, october 2005.

[Mak04] J. Makowsky : Algorithmic uses of the Feferman–Vaught Theorem, *Annals of Pure and Applied Logic* **126** (2004) 159–213

[Mak05] J. Makowsky : Coloured Tutte polynomials and Kauffman brackets for graphs of bounded tree-width, *Discrete Applied Maths.* **145** (2005) 276-290.

[Mak06a] J. Makowsky : From a Zoo to a Zoology: Descriptive Complexity for Graph Polynomials, In: A. Beckmann, et al. (eds.): *Logical Approaches to Computational Barriers, Second Conference on Computability in Europe*, *Lecture Notes in Computer Science*, **3988** (2006) 330-341.

[Mak06b] J. Makowsky, U. Rotics, I. Averbouch and B. Godlin, Computing graph polynomials on graphs of bounded clique-width, In: *Proceedings of WG06*, H. Bodlaender et al. (eds), *LNCS* 4271 (2006) 191-204

[Nob] S. Noble, Evaluating the Tutte polynomial for graphs of bounded tree-width, *Combin. Proba. Computing* **7** (1998) 307-321.

[Oum] S. Oum: Approximating Rank-Width and Clique-Width Quickly. *Proceedings WG 2005*, *LNCS* **3787** (2005) 49-58.

[Sok] A. Sokal: The multivariate Tutte polynomial (alias Potts model) for graphs and matroids. *Surveys in Combinatorics*, in Volume **327** of *London Math. Soc. Lec. Notes*, 2005, pp. 173-226.

[Whi] N. White (ed.), *Matroid applications*, Cambridge University Press, 1992.

## 8 Appendix : Tutte polynomial

In this section we apply our logical tools to the Tutte polynomial of matroids. We define a multivariate Tutte polynomial "better" than Sokal's. We first establish a lemma.

## 8.1 Enumerating polynomials

The *enumerating polynomial* of a set  $S \subseteq \mathcal{P}(V)$  is defined as  $Enum(S) = \sum_{A \in S} x_A$ . We write  $Enum_x(S)$  to specify the generic type  $x$  of indeterminates. (Hence  $Enum_y(S) = \sum_{A \in S} y_A$ ).

For a polynomial  $P$ , we let  $P^-$  (resp.  $P^+$ ) denote the polynomial obtained by replacing each indeterminate  $x$  (of any type) by  $x - 1$  (resp. by  $x + 1$ ).

If  $B \subseteq V$ , we let  $Sub(B) := \{A \mid A \subseteq B\}$ .

**Lemma A.1** : For every  $S \subseteq \mathcal{P}(V)$ , the following are equivalent :

- (1)  $S = Sub(B)$  for some  $B \subseteq V$ ,
- (2)  $Enum(S)^-$  is positive,
- (3)  $Enum(S)^- = x_B$  for some  $B \subseteq V$ .

**Proof** : (1)  $\implies$  (3)  $\implies$  (2) is straightforward. The set  $B$  in (3) is the same as in (1).

(3)  $\implies$  (1) : We note that  $Enum(S) = (Enum(S)^-)^+$  ; if  $Enum(S)^- = x_B$  then  $Enum(S) = x_B^+ = Enum(Sub(B))$  hence  $S = Sub(B)$ .

(2)  $\implies$  (3). By induction on the cardinality of  $V$ .

We let  $S \setminus a$  denote  $\{A - a \mid a \in A \in S\}$  and  $S - a = \{A \mid a \notin A \in S\}$ . We have thus

$$\begin{aligned} Enum(S) &= x_a \cdot Enum(S \setminus a) + Enum(S - a), \text{ and} \\ Enum(S)^- &= (x_a - 1) \cdot Enum(S \setminus a)^- + Enum(S - a)^-. \end{aligned}$$

It is positive by hypothesis. So is  $Enum(S \setminus a)^-$  (just look at monomials with  $x_a$ ), hence  $S \setminus a = Sub(B')$  for some  $B' \subseteq V - a$ .

Then  $Enum(S - a)^- - Enum(S \setminus a)^-$  is also positive. Since this is the case for  $Enum(S \setminus a)^-$  the same holds for  $Enum(S - a)^-$ .

Hence  $S - a = Sub(C)$  for some  $C \subseteq V - a$ . Hence

$Enum(S)^- = (x_a - 1)x_{B'} + x_C$ . That  $Enum(S)^-$  is positive implies  $B' = C$ . Hence  $S = Sub(B)$  where  $B = B' \cup \{a\}$  and  $Enum(S)^- = x_B$ .  $\square$

## 8.2 The Tutte polynomial of a matroid

We represent the Tutte polynomial of a matroid as a specialization of another multivariate MS-polynomial than the one defined by Sokal in [Sok] recalled in the introduction. We establish at the multivariate level that its coefficients are positive, with help of the notion of an *active element* with respect to a basis and a linear ordering.

We consider the Tutte polynomial for matroids but the application to graphs is immediate. Our reference is the book edited by N. White [Whi]. We refer to this book for the basic definitions on matroids and some results. A matroid  $M = (E, I)$  with base set  $E$  (we can think of  $E$  as the edge set of a graph) is considered

as a relational structure where  $I$  is a *set predicate* defining the independent sets. MS logic can be used to describe matroids (see [Hli03], [CouOum]). For instance, the bases of  $M$  are characterized by the MS formula  $\beta(X)$  :

$$I(X) \wedge \forall Z \cdot [\{X \subseteq Z \wedge I(Z)\} \implies Z \subseteq X].$$

The *rank polynomial* of  $M$  is defined as :

$$R(M) = \sum_{A \subseteq E} x^{r(M)-r(A)} y^{n(A)}$$

where  $r(M)$  is the rank of  $M$ ,  $r(A)$  is that of  $A$  and  $n(A) := |A| - r(A)$ . The Tutte polynomial is defined as :

$$T(M) := R(M)^- := \sum_{A \subseteq E} (x-1)^{r(M)-r(A)} (y-1)^{n(A)} .$$

Whereas  $R(M)$  is clearly positive, this is not obvious for  $T(M)$  from this definition.

We let  $\sigma$  be the substitution that replaces each  $x_a$  by  $x$  and each  $y_a$  by  $y$ . In our logical setting, the rank polynomial can be expressed as  $\sigma \circ \widehat{R}$ , for  $M$  linearly ordered in an arbitrary way where :

$$\widehat{R}(M) = \sum_{\varphi(A,C,D)} x_C y_D,$$

and the MS-formula  $\varphi(A,C,D)$  expresses the following conditions :

- (a)  $A \subseteq E$ ,
- (b) there exists a set  $Z$  such that  $Z \subseteq A$ ,  $Z$  is a maximal independent subset of  $A$ , that is lexicographically minimal with these properties,
- (c)  $C \subseteq E - A$  is lexicographically minimal such that  $Z \cup C$  is a base of  $M$ , and
- (d)  $D = A - Z$ .

Hence  $(C, D, Z)$  is associated in a unique way with  $A$  and  $|C| = r(M) - r(A)$ ,  $|D| = |A| - r(A) = n(A)$ . It follows that  $R(M) = \sigma(\widehat{R}(M))$ .

Hence, the rank polynomial is an MS-polynomial. For proving that the Tutte polynomial is positive, we will replace  $\widehat{R}$  by another multivariate polynomial, in such a way that we can deduce from Lemma A.1 that the Tutte polynomial is positive. Our main tool is the following proposition from [Bjo]. We need some definitions relative to an *ordered matroid*  $M = (E, I, \leq)$  (i.e.,  $E$  is linearly ordered).

For every base  $B$ , an element  $e$  of  $E - B$  is *externally active with respect to  $B$*  if it is the least element with respect to  $\leq$  of its *fundamental cycle* (the unique cycle included in  $B \cup \{e\}$ ). Their set is denoted by  $EA(B)$ . Dually, an element  $b$  of  $B$  is *internally active in  $B$*  if it is the least element of its fundamental cocycle. Their set is denoted by  $IA(B)$ .

**Proposition A.2** ([Bjo], Proposition 7.3.6) : For every ordered matroid  $M = (E, I, \leq)$  :

$$\mathcal{P}(E) = \uplus_{\beta(B)} \{A \mid B - IA(B) \subseteq A \subseteq B \cup EA(B)\}.$$

We recall that  $\uplus$  denotes a disjoint union and that  $\beta(B)$  expresses that  $B$  is a basis. Hence every subset  $A$  of  $E$  can be decomposed in a *unique way* as :

- (i)  $A = (A \cap B) \cup (A - B)$
- (ii) with  $A \cap B \supseteq B - IA(B)$  and  $A - B \subseteq EA(B)$ , for a unique base  $B$ .
- (iii) We let  $C = B - A$ . Hence  $C \subseteq IA(B)$ . Then  $r(A) = |A \cap B|$ . (See [Bjo] for the proof). It follows that  $|C| = r(M) - r(A)$ .
- (iv) Let  $D = A - B$ : then  $n(A) = |D|$ .

The monomial  $x_C y_D$  satisfies :

$$(v) \sigma(x_C y_D) = x^{r(M)-r(A)} y^{n(A)}.$$

We define :

$$\tilde{R}(M) = \sum_{\beta(B)} \sum_{\theta(B,C,D)} x_C y_D,$$

where  $\theta(B, C, D)$  expresses that  $C \subseteq IA(B)$  and  $D \subseteq EA(B)$ .

**Claim :**  $R(M) = \sigma(\tilde{R}(M))$ .

**Proof :** Let  $A \subseteq E$  and  $m(M, A) = x^{r(M)-r(A)} y^{n(A)}$  be the corresponding monomial in the expression of  $R(M)$  as  $\sum_{A \subseteq E} m(M, A)$ .

Let  $B$  be the unique basis satisfying conditions (i)-(ii). Let  $C$  and  $D$  be associated with  $A$  by (iii) and (iv). Then, by (v),  $m(M, A)$  is the image by  $\sigma$  of a unique monomial of  $\tilde{R}(M)$ .

Conversely, let  $x_C y_D$  be a monomial of  $\tilde{R}(M)$ . Let  $A = (B - C) \cup D$ . Conditions (i)-(iv) hold, hence  $(B, C, D)$  is the unique triple associated with  $A$  by these conditions, and thus  $\sigma(x_C y_D) = m(M, A)$ . This proves the claim.  $\square$

Note that for each basis  $B$ , the condition  $\theta(B, C, D)$  expresses that  $C$  and  $D$  can be chosen independently, with  $C \subseteq IA(B)$  and  $D \subseteq EA(B)$ . Hence  $R(M) = \sigma(\tilde{R}(M))$  where :

$$\tilde{R}(M) = \sum_{\beta(B)} Enum_x(Sub(IA(B))) \cdot Enum_y(Sub(EA(B))).$$

Let us define

$$\tilde{T}(M) = \sum_{\beta(B)} x_{IA(B)} y_{EA(B)}.$$

This is our alternative *multivariate Tutte polynomial for an ordered matroid*  $M$ . Clearly,

$$\begin{aligned} \tilde{R}(M)^- &= \\ &= \left\{ \sum_{\beta(B)} Enum_x(Sub(IA(B))) \cdot Enum_y(Sub(EA(B))) \right\}^- \\ &= \sum_{\beta(B)} Enum_x(Sub(IA(B)))^- \cdot Enum_y(Sub(EA(B)))^- \\ &= \sum_{\beta(B)} Enum_x(\{IA(B)\}) \cdot Enum_y(\{EA(B)\}) = \sum_{\beta(B)} x_{IA(B)} y_{EA(B)} \end{aligned}$$

$$= \tilde{T}(M).$$

We have :

$$R(M)^- = \sigma(\tilde{R}(M))^- = \sigma(\tilde{R}(M)^-) = \sigma(\tilde{T}(M)) = T(M).$$

Hence  $T(M)$  and  $\tilde{T}(M)$  are both positive. We have lifted at the multivariate level, the well-known identity :

$$\begin{aligned} T(M) &= \sum_{A \subseteq E} (x-1)^{r(M)-r(A)} (y-1)^{n(A)} \\ &= \sum_{\beta(B)} x^{|IA(B)|} y^{|EA(B)|} \\ &= \tilde{R}(M)^- \end{aligned}$$

because we have  $\tilde{T}(M) = \tilde{R}(M)^-$ .

*Computing and evaluating the Tutte polynomial.*

The polynomials

$$\tilde{R}(M) = \sum_{\beta(B)} \sum_{\theta(B,C,D)} x_C y_D, \text{ and } \tilde{T}(M) = \sum_{\beta(B)} x_{IA(B)} y_{EA(B)}$$

are thus MS-polynomials, because the notion of a basis, the sets  $IA(B)$  and  $EA(B)$ , the conditions  $\theta(B,C,D)$  are MS-expressible. So are  $R(M)$  and  $T(M)$ . Some consequences regarding their computations follow along the lines of Theorem 29.

This approach is developed by Makowsky in [Mak05] for graphs of bounded tree-width. (The application of Theorem 29 is not immediate because a linear ordering of the edge set must be incorporated to the relational structure in such a way it remains of bounded tree-width.) Polynomial algorithms that compute the Tutte polynomial of graphs of bounded tree-width are given in [And, Nob]; these algorithms do not use monadic second-order logic. However they use so-called "splitting" formulas which can be seen as particular cases of equalities (1) and (2) of Theorem 26, which are central for the proof of Theorem 29 (as pointed out in [Mak05]).

Gimenez et al. give a non-polynomial algorithm that computes the Tutte polynomial of graphs of bounded clique-width ([GHN]). Theorem 29 is not applicable in this case because edge set quantifications are necessary and  $MS_2$  logic does not fit with bounded clique-width.

For matroids of finite branch-width represented on finite fields, Hlineny gives a polynomial algorithm [Hli06] that follows the ideas of those of [And, Nob]. However MS properties of such matroids can be checked in polynomial time (see [Hli03]), hence the method of Theorem 29 may perhaps be applied also to these matroids.