# Computing Graph Polynomials on Graphs of Bounded Clique-Width

J.A. Makowsky[1], Udi Rotics[2], Ilya Averbouch[1], and Benny Godlin[13]

[1] Department of Computer Science
Technion–Israel Institute of Technology, Haifa, Israel
janos@cs.technion.ac.il
[2] School of Computer Science and Mathematics,
Netanya Academic College, Netanya, Israel,
rotics@mars.netanya.ac.il
[3] IBM Research and Development Laboratory, Haifa, Israel

**Abstract.** We discuss the complexity of computing various graph polynomials of graphs of fixed clique-width. We show that the chromatic polynomial, the matching polynomial and the two-variable interlace polynomial of a graph $G$ of clique-width at most $k$ with $n$ vertices can be computed in time $O(n^{f(k)})$, where $f(k) \leq 3$ for the inerlace polynomial, $f(k) \leq 2k + 1$ for the matching polynomial and $f(k) \leq 3 \cdot 2^{k+2}$ for the chromatic polynomial.

## 1 Introduction

In this paper[1] we deal with the complexity of computing various graph polynomials of a simple graph of clique-width at most $k$. Our discussion focuses on the univariate **characteristic** polynomial $P(G, \lambda)$, the **matching** polynomial $m(G, \lambda)$, the **chromatic** polynomial $\chi(G, \lambda)$, the multivariate **Tutte** polynomial $T(G, X, Y)$ and the **interlace** polynomial $q(G, XY)$, cf. [Bol99], [GR01] and [ABS04b]. All these polynomials are not only of graph theoretic interest, but all of them have been motivated by or found applications to problems in chemistry, physics and biology. We give the necessary technical definitions in Section 3.

Without restrictions on the graph, computing the characteristic polynomial is in **P**, whereas all the other polynomials are ♯**P** hard to compute. Clique-width is a parameter of graphs similar, but more flexible, than its related notion of tree-width, [CO00], [CMR01]. Tree-width plays an important rôle in parametrized complexity theory, as shown forcefully in the monograph of Downey and Fellows [DF99].

We assume the reader is familiar with the basic of complexity theory as given in [Pap94], and with the notion of tree-width of a graph. General background on tree-width may be found in [Die96].

---

[1] We report here some of the results obtained in the first author's seminar 238900 on graph polynomials, held in 2005 at the CS Department of the Technion.

We distinguish between the following upper bounds for the running time of algorithms with input size $n$ and a parameter $k$ of the input:

**Fixed parameter exponential time (FPEXP).** Runtime less than $2^{n^{c_1(k)}}$ with $c_1(k) \geq 1$.

**Fixed parameter subexponential time (FPSUBEXP).** Runtime less than $2^{c_2(k) \cdot n^{1-\epsilon(k)}}$.

**Fixed parameter polynomial time (FPPT).** Runtime less than $n^{c_3(k)}$.

**Fixed parameter tractable (FPT).** Runtime less than $c_4(k) \cdot n^d$.

**Polynomial time (P).** Runtime less than $O(n^d)$.

Here $d$ is independent of $n$ and $k$, whereas the other constants may depend on $k$. In this paper $k$ will be either the tree-width or the clique-width of the input graph $G$.

Our main results are summarized in the following theorem.

**Theorem 1** *There are algorithms such that, for a graph $G$ with $n$ vertices of clique-width at most $k$, given together with its $k$-expression,*

(i) *the interlace polynomial $q(G, X, Y)$ can be computed in time $O(n^d)$, where $d$ is constant independent of $n$ and $k$. Hence it is in **FPT** for $k$, cf. Proposition 3.*

(ii) *the matching polynomial $m(G, \lambda)$ can be computed in time $O(n^{f(k)})$, where $f$ is function independent of $n$ and linear in $k$. Hence it is in **FPPT** for $k$. cf. Theorem 4.*

(iii) *the chromatic polynomial $\chi(G, \lambda)$ can be computed in time $O(n^{f(k)})$, where $f$ is function independent of $n$ and simply exponential in $k$. Hence it is in **FPPT** for $k$. cf. Theorem 6.*

We summarize the known and new results in table 1.

| Polynomial | tree-width $k$ | clique-width $k$ |
|---|---|---|
| $P(G, \lambda)$ | in **P** | in **P** |
| $\chi(G, \lambda)$ | in **FPT** for $k$ [And98,Nob98] | in **FPPT** for $k$ NEW |
| $m(G, \lambda)$ | in **FPT** for $k$ [Mak04] | in **FPPT** for $k$ NEW |
| $q(G, X, Y)$ | in **FPT** for $k$ NEW | in **FPT** for $k$ NEW |
| $T(G, X, Y)$ | in **FPT** for $k$ [And98,Nob98] | in **FPSUBEXP** for $k$ [GHN05] |

**Table 1.** Overview of complexity

Note that in the case of the matching polynomial $f$ is linear in $k$, whereas for the chromatic polynomial it is exponential in $k$. For arbitrary graphs only the characteristic polynomial is in **P**. The others are all $\sharp$**P**-hard, [Val79], [ABS04b], [JVW90].

**Relevance of the results.** We report here on ongoing research into the complexity of computing graph polynomials on graphs of bounded clique-width. There are two ways of approaching the problem: Either one establishes that one can apply [Mak04, Theorem 6.6], which can be far from obvious, or use explicit methods. For the interlace polynomial we indeed use the first apporach. For counting problems the explicit approach was applied to the chromatic number in [KR03], and to ♯**SAT** in [FMR06]. For the Tutte polynomial it was applied in [GHN06]. To the best of our knowledge no other work on the complexity of graph polynomials on graph classes of bounded clique-width was published. Our explicit results show remarkable differences concerning the various polynomials. Whether this is inherent or just due to the absence of more sophisticated algorithms remains open. We suspect that the differences are inherent.

## 2   Clique-width of graphs

The notion of clique-width was introduced in [CER93] and developed more systematically in [CO00]. $k$-graphs are graphs with vertices labeled with possibly one out of $k$ many labels. The class of graphs $CW(k)$ of clique-width at most $k$ is defined inductively. Singleton $k$-graphs are in $CW(k)$. $CW(k)$ is closed under disjoint union $\oplus$, relabeling of the label $i$ by the label $j$, denoted by $\rho_{i \to j}$, and, for $i \neq j$, edge creation $\eta_{i,j}$ where all the vertices labeled $i$ are connected to all the vertices labeled $j$. The clique-width of a graph $G$ is the minimum $k$ such that $G \in CW(k)$. A $k$-expression $t$ is a term built using the terms for singleton $k$-graphs, disjoint union, relabeling and edge creation. We denote by $val(t)$ the $k$-graph described by $t$. Courcelle and Olariu in [CO00] showed that the clique-width of graphs of tree-width at most $k$ are in $CW(2^{k+1}+1)$. Therefore, any class of graphs of bounded tree-width is automatically of bounded clique-width. The best known bound is due to [CR05]. Given a graph $G$, computing its tree-width is **NP**-hard, [ACP87]. The same was recently shown to be true for clique-width [FRRS05]. Deciding whether $G$ has tree-width at most $k$ is in **FPT** for $k$. For clique-width this is not known to be true. However, S. Oum and P. Seymour, [OS05], showed

**Theorem 1 (S. Oum and P. Seymour, 2004).** *There is polynomial time algorithm, which for fixed $k$, decides whether the graph has clique-width at least $k + 1$ or else outputs a $k_1 = (2^{3k+2} - 1)$-expression for the graph showing that its clique-width is at most $k_1$.*

In general, it seems that finding an explicit bound for the clique-width is a more complicated task than finding a bound for the tree-width.

## 3   Graph polynomials

Let $\mathcal{G}$ be the class of graphs $G = (V, E)$ without loops and multiple edges. Let $\mathcal{R}$ be a ring and $\bar{X}$ be a (not necessarily finite) set of indeterminates. A *graph polynomial* is a function

$$p : \mathcal{G} \to \mathcal{R}[\bar{X}]$$

such that for isomorphic graphs $G_1 \simeq G_2$ we have $p(G_1) = p(G_2)$.

There are plenty of graph polynomials which have been discussed in the literature, although no systematic treatment on graph polynomials in general is available. To put our results into perspective we discuss briefly several classical graph polynomials, the *chromatic polynomial* $\chi(G, \lambda)$, the *characteristic polynomial* $P(G, \lambda)$, the *acyclic generating matching polynomials* $m(G, \lambda)$ *and* $g(G, \lambda)$ and the *Tutte polynomial* $T(G, X, Y)$.

**The chromatic polynomial** . Let $\chi(G, \lambda)$ denote the number of proper vertex colorings of $G$ with at most $\lambda$ many colors. G. Birkhoff, [Bir12], observed in 1912 that $\chi(G, \lambda)$ is, for a fixed graph $G$, a polynomial in $\lambda$, which is now called the *chromatic polynomial of* $G$. The chromatic polynomial is the oldest graph polynomial to appear in the literature, and since then a substantial body of knowledge about the chromatic polynomial of graphs and its applications has been accumulated. The recent book by F.M. Dong, K.M. Koh and K.L. Teo [DKT05] gives an excellent and extensive survey. One of the surprising facts is a theorem of R.P. Stanley, [Sta73], which states that $\chi(G, -1)$ is the number of acyclic orientations of $G$.

**The Tutte polynomial.** Interesting generalizations of the chromatic polynomial were introduced by H. Whitney in 1932 and Tutte in 1947. The most prominent among them is now called the *Tutte polynomial* $T(G, X, Y)$ which is a two variable polynomial from which the chromatic polynomial can be obtained via a simple substitution and multiplication with a prefactor. For a modern exposition the reader is referred to [Bol99, chapter X], [Wel93]. For this paper we do not need a definition of the Tutte polynomial.

Other univariate graph polynomials were introduced after 1955, often first motivated by problems from chemistry and physics.

**The characteristic polynomial of a graph** $G$**,** denoted by $P(G, \lambda)$ is the characteristic polynomial of the adjacency matrix $M_G$ of the graph $G$, $P(G, \lambda) = \det(\lambda \cdot \mathbf{1} - M_G)$ and is completely determined by the eigenvalues of $M_G$, which are all real, as the matrix is symmetric.

**The matching polynomials.** The *acyclic polynomial of* $G$ is the polynomial $m(G, \lambda) = \sum_k (-1)^k \cdot m_k(G) \cdot \lambda^{n-2k}$, where the coefficients $m_k(G)$ count $k$-matchings. A chemical point of view of these polynomials is given in [CDS95] and [Tri92], where also algorithmic aspects are touched. A close relative of the acyclic polynomial is the *generating matching polynomial of a graph* $G$ $g(G, \lambda) = \sum_k m_k(G)\lambda^k$ where $m(G, \lambda) = \lambda^n g(G, (-\lambda^{-2}))$. An excellent survey on these two matching polynomials may be found in [LP86, Chapter 8.5]. We shall refer to both as *matching polynomials*.

**The interlace polynomials.** The interlace polynomials were introduced in [ABS00,ABS04a,ABS04b]. and further studied in [AvdH04]. The most general

version is a two-variable version from [ABS04b] given by

$$q(G, X, Y) = \sum_{S \subseteq V(G)} (X - 1)^{r_2(G[S])} (Y - 1)^{|V| - r_2(G[S])}$$

where $G[S]$ is the subgraph induced by $S$ and $r_2(G)$ is the matrix rank over $\mathbb{Z}_2$ of the adjacency matrix of $G$. The polynomial introduced in [ABS00] can be written as

$$q_N(G, Y) = q(G, 2, Y) = \sum_{S \subseteq V(G)} (Y - 1)^{|V| - r_2(G[S])}$$

The evaluation $q(G, 1, 2)$ counts the number of independent sets of a graph, [ABS04b].

**Relationship between the polynomials.** It is well known that the chromatic polynomial can be obtained from the Tutte polynomial by a simple substitution and multiplication with a polynomial time computable graph invariant. It is open whether such a reduction exists between the remaining polynomials (not including the trivial relationships with the characteristic polynomial). But for each two polynomials one can find pairs of graphs $G_1, G_2$ for which one polynomial gives the same value and for the other it gives different values.

## 4  Complexity of computing the graph polynomials

**The general situation.** It is natural to ask how difficult it is to compute the various graph polynomials. We look at two versions of this problem:

**Coefficents:** Given a graph $G$, compute all the coefficients of the polynomial $p(G, \lambda)$.

**Evaluation:** Given a graph $G$, evaluate the polynomial $p(G, \lambda)$ for a fixed $\lambda = \lambda_0 > 0$ in the underlying ring $\mathcal{R}$.

If the ring $\mathcal{R}$ is the ring of integers $\mathbb{Z}$ these problems are well defined in the Turing model of computations. For arbitrary rings it is best to work in the unit-cost model of computation as described in, say, [BCSS98].

Clearly, if we can evaluate a polynomial efficiently, also its coefficents can be computed efficiently. Evaluating the polynomial $\chi(G, \lambda)$ for integers $\lambda \geq 3$ is $\sharp\mathbf{P}$-complete, [Val79]. For the best known algorithms, see [FK03]. Even if restricted to planar graphs, counting the number of 3-colorings or the number of acyclic orientations, i.e. evaluating the polynomial $\chi(G, \lambda)$ at $\lambda = 3$ and $\lambda = -1$, is known to be $\sharp\mathbf{P}$-hard [VW92]. This also makes evaluating the Tutte polynomial $\sharp\mathbf{P}$-hard. The same is true for the acyclic polynomial due to its connection to counting matchings, and for the interlace polynomial, due to its counting of independent sets, cf. [Val79]. However, it remains open, whether evaluating $q_N(G, Y)$ is $\sharp\mathbf{P}$-hard, cf. [ABS04b]. In their remarkable paper, [JVW90], F. Jaeger, X. Vertigan and D. Welsh, have characterized completely the points $(a, b)$ in the complex

plane, where evaluating the Tutte polynomial $T(G, a, b)$ is difficult for arbitrary graphs.

Computing the coefficients for univariate polynomials of degree $d$ can be done in polynomial time from $d + 1$ evaluations at different points. Computing the coefficients for multivariate polynomials is in general more complicated. The the coefficients of the characteristic polynomial are computable in polynomial time using classical algorithms for the determinant of a matrix.

## 5  Bounded tree-width

J. Oxley and D. Welsh [OW92] also noted that the Tutte polynomial for series parallel graphs, which are graphs of tree-width at most 2, can be computed in polynomial time. This was extended to arbitrary fixed tree-width $k$ independently by A. Andrzejak [And98] and S. Noble [Nob98], and therefore also holds for the chromatic polynomial. Actually, they showed that computing the Tutte polynomial is in **FPT** on graph classes of tree-width at most $k$ with computation time roughly $f(k)n^3$ where $f(k)$ is simply exponential in $k$ and the tree-decomposition of the graph is given in advance.

The same result also follows from a more general method presented in [Mak04], which also covers the acyclic polynomial and a wide range of other graph polynomials where summations is restricted to families of subsets of edges which are definable in Monadic Second Order Logic, including, using Proposition 3 below, the interlace polynomial. However, even when the tree decomposition is given in advance, $f(k)$ here would be at least doubly exponential.

For the matching polynomial we have improved this.

**Theorem 2** *If the graph $G$ is given together with its $k$-tree decomposition then the matching polynomial can be computed in time $O(2^{3k} \cdot n^3)$ where the remaining constants are independent of $k$ and $n$.*

*Proof (Sketch).* As in [CO00], we translate a $k$-tree decomposition of the input graph into a $k$-expression, but we note that the $\eta_{i,j}$'s are only applied to sets with label $i$ and $j$ which are smaller than $k$. Then we proceed as in the proof of Theorem 4 below. □

Similar results for polygraphs, which are special case of graph classes of bounded path-width, were obtained already in [BGMP86]. For the Tutte polynomial, and hence for the chromatic polynomial, similar bounds can be obtained from [And98,Nob98].

## 6  Bounded clique-width: the interlace polynomial

In the same paper [Mak04], in Theorem 6.6 it is shown that, in combination with the work of P. Seymour and S. Oum [OS05], graph polynomials, where summations are restricted to families of subsets of vertices which are definable in Monadic Second Order Logic, are in **FPT** for graph classes of clique-width at most $k$. To apply this to the interlace polynomial it suffices to show

**Proposition 3** *The interlace polynomial can be written as*

$$\sum_{A:\phi(A,B,C)} \prod_{u:u\in B} U \prod_{v:v\in C} V$$

*where $\phi(A,B,C)$ is a formula with free monadic variables $A,B,C$ in* **MSOL** *with a parity quantifier. In other words, it is a* $\mathbf{C_2}$**MSOL**-*definable graph polynomial.*

*Proof (Sketch).* The interlace polynomial is defined as

$$q(G,X,Y) = \sum_{S\subseteq V(G)} (X-1)^{r_2(G[S])}(Y-1)^{|V|-r_2(G[S])}$$

First we substitute $X - 1 = U$ and $Y - 1 = V$. To get the formula $\phi$, we formalize vectors of the adjacency matrix by unary predicates. This can be done in **MSOL**. To compute the rank in $GF(2)$ we use the fact that in $GF(2)$ a sum of 1's vanishes iff it consists of an even number of summands. For this part we need a counting quantifier $\mathbf{C_2}$. □

A very similar argument can be found in [CO06].

Theorem 1 (i) now follows from [Mak04,OS05]. However, this method does not apply to the chromatic polynomial, the Tutte polynomial and the matching polynomials.

## 7 Bounded clique-width: the matching polynomial

A more explicit version of Theorem 1(ii), combined with Theorem 1 is the following:

**Theorem 4** *The matching polynomial $m(G,\lambda)$ of a graph $G$ with $n$ vertices of clique-width at most $k$ can be computed in polynomial time. More precisely, in time $O(n^{\alpha(k)})$, with $\alpha(k) = O(2k+1)$ if a $k$-expression for the graph $G$ is given. Hence it is in* **FPPT**. *Otherwise, $\alpha(k)$ is simply exponential in $k$.*

For the proof we first restrict our attention to $k$-expression which are *irredundant*. Then we introduce a set of auxiliary polynomials, the *constrained matching polynomials*. We use the dynamic programming approach to compute the auxiliary polynomials. Finally, we piece everything together.

**Irredundant $k$-expressions.** Let $K = 1, 2, ..., k$ be a set of $k$ labels, and $T(K)$ be the set of all the possible $k$-expressions using labels of $K$. A $k$-expression is called *irredundant* if for every of its subexpression of the form $\eta_{i,j}(t')$ no vertex labeled $i$ is adjacent to vertex labeled $j$ in $val(t')$. According to [CO00], every graph built inductively from a $k$-expression $t \in T(K)$, can also be built from an irredundant $k$-expression $t' \in T(K)$. To proceed, let $G$ be a graph, and $t$ be its irredundant $k$-expression. We want to compute its matching polynomial $m(G,\lambda)$. We denote by $tree(t)$ the parse-tree of $t$. We shall use a set of auxiliary polynomials.

**Constrained matching polynomials $cm_F(H, \lambda)$.** Let $H(V, E)$ be a labeled graph with labels $1, 2, ..., k$. We denote by $P_i$ the set of vertices $v \in V$ labeled $i$. Let $F = (f_1, f_2, ..., f_k)$ be a vector of $k$ non-negative integers $0 \leq f_i \leq n$, where $n = |V|$. We will denote by $cm_F^l(H)$ the number of $l$-matchings, which leave unused exactly $f_i$ vertices labeled $i$, for every label $1 \leq i \leq k$. The **constrained matching polynomial** is defined by:

$$cm_F(H, \lambda) = \sum_{l=0}^{\frac{n}{2}} cm_F^l(H) \lambda^l$$

Obviously, the size of suitable matchings is uniquely defined by $F$, so actually $cm_F(H, \lambda)$ is a monomial. Our set of auxiliary functions is: $\{cm_F(H, \lambda)\}$.

**The algorithm.**

**Singletons labeled $i$:** If $F = (0, 0, ...0, 1, 0, ..., 0)$ with a single 1 at the $i$-th position, then $cm_F(H, \lambda) = 1$, otherwise $cm_F(H, \lambda) = 0$.

$\oplus$**:** Assume $H = H_1 \oplus H_2$. Since the graphs $H_1$ and $H_2$ are disjoint, it is easy to check that

$$cm_F(H) = \sum_{F_1} cm_{F_1}(H_1) \cdot cm_{(F-F_1)}(H_2) \tag{1}$$

$F - F_1$ is vector subtraction. If $F - F_1$ has a negative coordinate we put $cm_{(F-F_1)}(H_2) = 0$.

$\rho_{i \to j}$**:** Assume $H = \rho_{i \to j}(H_1), i \neq j$. Then we have:

$$cm_F(H) = \sum_{F':(F,F') \models \varphi} cm_{F'}(H_1) \tag{2}$$

where

$$\varphi = \left( \forall_{l:1 \leq l \leq k} : f_l = \begin{cases} f_i' + f_j' & if \quad l = j \\ 0 & if \quad l = i \\ f_l' & otherwise \end{cases} \right) \tag{3}$$

$\eta_{i,j}$**:** Assume $H = \eta_{i,j}(H_1), i \neq j$. Since $t$ is an irredundant $k$-expression, this operation adds new edges between all the vertices labeled $i$ and all the vertices labeled $j$. Any matching of $H$ consists of some matching of $H_1$ and (optionally) some new edges. If such a matching includes $q$ new edges, then it chooses two sets $A$ and $B$ and a bijection between $A$ and $B$, where $A$ consists of $q$ of $f_i$ free vertices labeled $i$, and $B$ consists of $q$ of $f_j$ free vertices labeled $j$. There are $q!$ many such bijections. Using the definition of $cm_F$ and counting the suitable matchings of the left and the right side of the equation we get:

$$cm_F(H) = \sum_{q, F' \models \varphi} \lambda^q \cdot cm_{F'}(H_1) \cdot \binom{f_i'}{q} \cdot \binom{f_j'}{q} \cdot q! \tag{4}$$

where

$$\varphi = \left( \forall_{l:1 \le l \le k} : f_l = \begin{cases} f_i' - q \ if \ \ l = i \\ f_j' - q \ if \ \ l = j \\ f_l' \quad\quad\ otherwise \end{cases} \right) \tag{5}$$

For the inductive computation of $m(G,\lambda)$ we observe that for different $F_1$ and $F_2$ the sets of matchings counted by $cm_{F_1}(H,\lambda)$ and $cm_{F_2}(H,\lambda)$ are disjoint. Furthermore, every matching of $H$ is counted by some $cm_F(H,\lambda)$. Hence, the matching polynomial of graph $G$ can be obtained by summation over $F$:

$$m(G,\lambda) = \sum_F cm_F(H,\lambda) \tag{6}$$

**Complexity analysis.**

(i) We initialize up to $(n^k)$ auxiliary monomials by 0 or 1.
(ii) For the disjoint union $\oplus$ we sum over all possible $F_1$, which is $O(n^k)$, for all the $(n^k)$ auxiliary monomials. This gives $O(n^{2k})$ steps.
(iii) For $\rho$ we sum over $F_1$, but the only free value is the $j$'th field of the vector ($O(n)$). This give, for $(n^k)$ auxiliary monomials, $O(n^{k+1})$ steps.
(iv) For $\eta$ we sum over $q$, which uniquely defines $F'$. We get, for $(n^k)$ auxiliary monomials, $O(n^{k+1})$ steps.
(v) Finally, we sum of all the auxiliary monomials, which gives $O(n^k)$ steps.

The number of times each operation is applied depends on the term $t$. $\oplus$ can appear at most $n$ times in $t$ because every time the number of vertices in $val(t)$ grows at least by 1. $\eta$ cannot be applied more than $n^2$ times, because it adds edges every time. $\rho$ can appear at most $n * k$ times for sequences of recolorings without repetitions. Summation of all the monomials is performed once. Hence we get that the total time complexity of the algorithm is bounded by

$$O(n \cdot (n^k) + n \cdot (n^{2k}) + n^2 \cdot (n^{k+1}) + n \cdot k \cdot (n^{k+1}) + (n^k)) = O(n^{2k+1}).$$

## 8 Bounded clique-width: the chromatic polynomial

For clique-width at most 2, which are the cographs, cf. [CO00], we have, using an observation of N. Biggs, [Big93, Chapter 9]:

**Theorem 5** *The chromatic polynomial of cographs with $n$ vertices can be computed in polynomial time.*

D. Kobler and U. Rotics [KR03] showed that the **chromatic number** of a graph with $n$ vertices of clique-width at most $k$ given as a $k$-expression can be computed in polynomial time $O(n^{\alpha(k)})$, with $\alpha(k) = O(2^k)$.

A more explicit version of Theorem 1(iii) combined with Theorem 1 is the following:

**Theorem 6** *The chromatic polynomial $\chi(G,\lambda)$ of a graph $G$ with $n$ vertices of clique-width at most $k$ can be computed in polynomial time. More precisely, in time $O(n^{\alpha(k)})$, with $\alpha(k) = O(2^k)$ if a $k$-expression for the graph $G$ is given. Otherwise, $\alpha(k)$ is doubly exponential in $k$.*

**Overview of the proof.** The value of the chromatic polynomial $\chi(G, \lambda)$ gives the number of different colorings of a graph $G$ using *at most* $\lambda$ colors. We denote by $\overline{\chi}(G, \lambda)$ the number of different colorings of a graph $G$ using *exactly* $\lambda$ colors. We say that two colorings $c$ and $d$ of $G$ are *isomorphic* if the set of sets of vertices induced by the colors in $c$ is equal to the set of sets of vertices induced by the colors in $d$. We denote by $\overline{\psi}(G, \lambda)$, the number of non-isomorphic colorings of a graph $G$ using exactly $\lambda$ colors.

The input to our algorithm is a graph $G$ together with a $k$-expression $t$ defining $G$. We denote by $K$ the set of labels $\{1, ..., k\}$ in the $k$-expression. The $k$-expression contains operations $\oplus, \eta_{i,j}$ and $\rho_{i \to j}$. The algorithms will traverse $tree(t)$ from bottom to top. It constructs at each step the labeled graph $H$ corresponding to a subtree of $tree(t)$ scanned so far, and keeps track of the labels and the number of their corresponding colorings.

If we wanted to compute $\chi(G, \lambda)$ directly using the $k$-expression $t$, the only difficult case would be $\eta_{i,j}$. However, we shall compute a different quantity, $num\text{-}cols(N, H)$, defined below. It turns out that the inductive computation of $num-cols(N, H)$ will be easy in the case of $\eta_{i,j}$ and $\rho_{i \to j}$ but a bit more involved in the case of the disjoint union $\oplus$.

**Introducing $num\text{-}cols(N, H)$.** Let $c$ be a coloring of a labeled graph $H$. The type of a color $i$ of $c$ is defined as the set of labels $B \subseteq K$, such that label $l$ belongs to $B$ if and only if there is a vertex of $H$ having label $l$ and color $i$. The type of the coloring $c$ is defined by counting for each $B \subseteq K$, the number of colors of type $B$ in $c$. In other words, the type of the coloring $c$ is defined as an array $N$ of size $2^k$, such that for every set of labels $B \subseteq K$, $N[B]$ contains the number of colors in $c$ of type $B$.

For an array $N$ we denote by $num\text{-}cols(N, H)$, the number of non-isomorphic colorings of $H$ which are of type $N$, and by $total(N)$ the sum of all the values of $N$. The information we keep is for each array $N$ and for each $H$ the value of $num\text{-}cols(N, H)$. Using this notation it is easy to see that

$$\overline{\psi}(G, \lambda) = \sum_{N: total(N) = \lambda} num\text{-}cols(N, G) \tag{7}$$

**Computing $num\text{-}cols(N, H)$ inductively.** Computing $num\text{-}cols(N, H)$, when $H$ is a singlton is straightforward. When $H = \eta_{i,j}(H_1)$ or when $H = \rho_{i \to j}(H_1)$ it is easy to see how to compute $num\text{-}cols(N, H)$, from $num\text{-}cols(N, H_1)$. We now consider the more complicated case when $H = H_1 \oplus H_2$.

Let $d$ and $e$ be colorings of $H_1$ and $H_2$, respectively using disjoint sets of colors. From these two colorings we can obtain colorings of $H_1 \oplus H_2$, by merging some of the colors of $d$ and $e$ into the same color. The colors merged are idenitifed by a set of pairs, denoted by $M$, where the pair $(i, j)$ belongs to $M$ if and only if color $i$ of $d$ is merged into color $j$ of $e$. The set of all possible merges for colorings $d$ and $e$ is senoted by $Merges(d, e)$. The coloring of $H$ obtained from colorings $d$ and $e$ using set of merges $M$ is denoted by $coloring(d, e, M)$.

For an array $N$ we denote by $Colorings(d, e, N)$ the set of all non-isomorphic colorings of $H$ which of type $N$ and are obtained from the colorings $d$ and $e$ using a set of merges in $Merges(d, e)$.

**Lemma 7** *For every four colorings $d, d', e, e'$ such that colorings $d$ and $d'$ are of the same type and the colorings $e$ and $e'$ are of the same type and for every array $N$, the number of non-isomorphic colorings in $Colorings(d, e, N)$ is equal to the number of non-isomorphic colorings in $Colorings(d', e', N)$.*

Note that Lemma 7 holds also when $d$ and $d'$ are colorings of different graphs and $e$ and $e'$ are colorings of different graphs.

Let $d$ and $e$ be any two colorings of $H_1$ and $H_2$ of type $N_1$ and $N_2$ respectively. We define $f(N_1, N_2, N) = |Colorings(d, e, N)|$. By Lemma 7 the value of $f(N_1, N_2, N)$ does not depend on the specific colorings $d$ and $e$ and on the graphs $H_1$ and $H_2$ colored by the colorings $d$ and $e$, respectively. Thus, the value $f(N_1, N_2, N)$ for each triple of arrays $N_1, N_2$ and $N$ can be calculated in a table before the algorithm starts. Using this notation we have

$$num\text{-}cols(N, H) = \sum_{N_1, N_2} num\text{-}cols(N_1, H_1) \cdot num\text{-}cols(N_2, H_2) \cdot f(N_1, N_2, N) \quad (8)$$

Formula 8 gives us the inductive step in the calculation $num\text{-}cols(N, H)$. Using Formula 7 we now can compute $\bar{\psi}(G, \lambda)$. Finally we have

$$\chi(G, \lambda) = \sum_{\alpha=1}^{\lambda} \bar{\psi}(G, \alpha) \cdot \alpha!.$$

**Complexity analysis.** The number of different arrays is at most $(\lambda + 1)^{2^k}$, since for each array $N$ and each $B \subseteq K$ the number of possible different values of $N[B]$ is at most $\lambda + 1$.

To analyze the complexity of the algorithm we consider the time taken by the $\oplus$ operations, since the other operations take less time. It can be shown that the routine which calculates all the values of $f(N_1, N_2, N)$ before the algorithm starts takes at most $O(2^{k+1} * (\lambda + 1)^{3*2^k})$ time which can be considered as $O(\lambda^{3*2^k})$, since $k$ is assumed to be a fixed constant.

For each array $N$, the value of $num\text{-}cols(N, H)$, evaluated by Formula 8 is achieved by considering all pairs of arrays $N_1, N_2$ and for each such pair evaluating two multiplications. Thus, the total time for calculating $num\text{-}cols(N, H)$ is at most $O(\lambda^{2*2^k})$. Repeating this calculation for every array $N$ we obtain that total time taken in calculating $F(H)$ on behalf of one $\oplus$ operation is at most $O(\lambda^{3*2^k})$. Since there are at most $n$ $\oplus$ operations the total complexity of the algorithm is at most $O(n * (\lambda^{3*2^k}))$. Putting $n = \lambda$, evaluating sufficiently often, and interpolating now gives Theorem 6.

By Stanley's Theorem, [Sta73], $\chi(G, -1)$ is the number of acyclic orientations of $G$. Hence we have:

**Corollary 8** *The number of acyclic orientations of a graph of clique-width at most $k$ can be computed in polynomial time.*

**Comparison with the Tutte polynomial.** O. Gimenez, P. Hliněný and M. Noy [GHN06] showed that the Tutte polynomial on graphs of clique-width at most $k$ given as a $k$-expression can be computed in time $2^{O(n^{1-\epsilon})}$ with $\epsilon = \frac{1}{k+2}$. This does not exclude that computing the Tutte polynomial could be $\sharp\mathbf{P}$-hard on graphs of bounded clique-width. But, together with Theorem 6, it excludes showing this via a reduction to the chromatic polynomial.

## 9 Conclusions and open problems

We have done a first step into analyzing the complexity of prominent graph polynomials on graphs of bounded clique width. All our polynomials are fixed parameter tractable (are in **FPT**) on graphs of tree-width at most $k$. However, on graphs of clique-width at most $k$ their complexity seems to differ. It is a challenge for future research to establish whether this is really so.

**Comparing the complexities.** The complexity of computing the chromatic polynomial for graphs $G$ of clique-width at most $k$, given the $k$-expression of $G$, is determined by the dimension of $N$, which is at least as numerous as the power set of the labels, i.e. $2^k$.

In contrast to this, in the case of the matching polynomial, the complexity is determined by the dimension of $F$, which is equal to $k$.

For the Tutte polynomial we could proceed like for the matching polynomial, but the analogue to the vector $F$ then contains values for each label and for each connected component of the spanning subgraphs. However, the number of components may equal the number of vertices of the graph. It is not clear how to avoid this and how to get an $F$ the size of which depends only on $k$.

**Open problems.**

 (i) Find explicit constants in the case of the interlace polynomial.
 (ii) Is computing the matching polynomial or the chromatic polynomial from $k$-expressions for graphs of clique-width at most $k$ in **FPT** for $k$?
(iii) Can the chromatic polynomial be computed in time comparable to the time needed for the matching polynomial?
(iv) Is computing the Tutte polynomial of graphs of bounded clique-width in **FPPT**, or even in **FPT** for $k$?
 (v) How can one generalize the underlying method for other graph polynomials. Although the ideas of [KR03] also helped in computing the matching polynomial, it is not yet clear, how to formulate a general principle.
(vi) Can one show, using the methods of [DF99], that computing the matching polynomial, chromatic polynomial or Tutte polynomial is not in **FPT** for graphs of clique-width at most $k$, for some $k$?

# References

[ABS00]   R. Arratia, B. Bollobas, and G.B. Sorkin. The interlace polynomial: a new graph polynomial. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Mathematics*, pages 237–245, 2000.

[ABS04a]  R. Arratia, B. Bollobas, and G.B. Sorkin. The interlace polynomial: a new graph polynomial. *Journal of Combinatorial Theory, Series B*, 92:199–233, 2004.

[ABS04b]  R. Arratia, B. Bollobas, and G.B. Sorkin. A two-variable interlace polynomial. *Combinatorica*, 24.4:567–584, 2004.

[ACP87]   S. Arnborg, D.G. Corneil, and A. Proskurowski. Complexity of finding embedding in a k–tree. *SIAM. J. Algebraic Discrete Methods*, 8:277–284, 1987.

[And98]   A. Andrzejak. An algorithm for the Tutte polynomials of graphs of bounded treewidth. *Discrete Mathematics*, 190:39–54, 1998.

[AvdH04]  M. Aigner and H. van der Holst. Interlace polynomials. *Linear Algebra and Applications*, 377:11–30, 2004.

[BCSS98]  L. Blum, F. Cucker, M. Shub, and S. Smale. *Complexity and real computation.* Springer-Verlag, New York, 1998.

[BGMP86]  D. Babić, A. Graovac, B. Mohar, and T. Pisanski. The matching polynomial of a polygraph. *Discrete Applied Mathematics*, 15:11–24, 1986.

[Big93]   N. Biggs. *Algebraic Graph Theory, 2nd edition.* Cambridge University Press, 1993.

[Bir12]   G.D. Birkhoff. A determinant formula for the number of ways of coloring a map. *Annals of Mathematics*, 14:42–46, 1912.

[Bol99]   B. Bollobás. *Modern Graph Theory.* Springer, 1999.

[CDS95]   D.M. Cvetković, M. Doob, and H. Sachs. *Spectra of graphs.* Johann Ambrosius Barth, 3 edition, 1995.

[CER93]   B. Courcelle, J. Engelfriet, and G. Rozenberg. Handle-rewriting hypergraph grammars. *J. Comput. System Sci.*, 46:218–270, 1993.

[CMR01]   B. Courcelle, J.A. Makowsky, and U. Rotics. On the fixed parameter complexity of graph enumeration problems definable in monadic second order logic. *Discrete Applied Mathematics*, 108(1-2):23–52, 2001.

[CO00]    B. Courcelle and S. Olariu. Upper bounds to the clique–width of graphs. *Discrete Applied Mathematics*, 101:77–114, 2000.

[CO06]    B. Courcelle and S. Oum. Vertex-minors, monadic second-order logic, and a conjecture by Seese. *Journal of Combinatorial Theory, Series B*, xx:xx–xx, 2006.

[CR05]    D. G. Corneil and U. Rotics. On the relationship between clique-width and treewidth. *SIAM J. Comput.*, 34(4):825–847, 2005.

[DF99]    R.G. Downey and M.F Fellows. *Parametrized Complexity.* Springer, 1999.

[Die96]   R. Diestel. *Graph Theory.* Graduate Texts in Mathematics. Springer, 1996.

[DKT05]   F.M. Dong, K.M. Koh, and K.L. Teo. *Chromatic polynomials and chromaticity of graphs*. World Scientific, 2005.

[FK03]    M. Fürer and S. P. Kasiviswanathan. Algorithms for counting 2-SAT solutions and colorings with applications. *Electronic Colloquium on Computational Complexity*, 1:R 33, 2003.

[FMR06]   E. Fischer, J.A. Makowsky, and E.V. Ravve. Counting truth assignments of formulas of bounded tree width and clique-width. *Discrete Applied Mathematics*, xx:xx–xx, 2006.

[FRRS05]  M.R. Fellows, F.A. Rosamond, U. Rotics, and S. Szeider. Proving NP-hardness for clique width. *ECCC*, xx:xx–yy, 2005.

[GHN05]   O. Giménez, P. Hliněný, and M . Noy. Computing the Tutte polynomial on graphs of bounded clique-width. In *Graph Theoretic Concepts in Computer Science, WG 2005*, volume 3787 of *Lecture Notes in Computer Science*, pages 59–68, 2005.

[GHN06]   O. Giménez, P. Hliněný, and M . Noy. Computing the Tutte polynomial on graphs of bounded clique-width. *XXX*, xx:xx–yy, 2006.

[GR01]    C. Godsil and G. Royle. *Algebraic Graph Theory*. Graduate Texts in Mathematics. Springer, 2001.

[JVW90]   F. Jaeger, D.L. Vertigan, and D.J.A. Welsh. On the computational complexity of the Jones and Tutte polynomials. *Math. Proc. Camb. Phil. Soc.*, 108:35–53, 1990.

[KR03]    D. Kobler and U. Rotics. Edge dominating set and colorings on graphs with fixed clique-width. *Discrete Applied Mathematics*, 126:197–221, 2003.

[LP86]    L. Lovasz and M. Plummer. *Matching Theory*. North Holland, 1986.

[Mak04]   J.A. Makowsky. Algorithmic uses of the Feferman-Vaught theorem. *Annals of Pure and Applied Logic*, 126:1–3, 2004.

[Nob98]   S.D. Noble. Evaluating the Tutte polynomial for graphs of bounded tree-width. *Combinatorics, Probability and Computing*, 7:307–321, 1998.

[OS05]    S. Oum and P. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Ser. B*, xx(x):xx–yy, 2005.

[OW92]    J.G. Oxley and D.J.A. Welsh. Tutte polynomials computable in polynomial time. *Discrete Mathematics*, 109:185–192, 1992.

[Pap94]   C. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.

[Sta73]   R. P. Stanley. Acyclic orientations of graphs. *Discrete Mathematics*, 5:171–178, 1973.

[Tri92]   N. Trinajstić. *Chemical graph theory*. CRC Press, 2 edition, 1992.

[Val79]   L.G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

[VW92]    D.L. Vertigan and D.J.A. Welsh. The computational complexity of the Tutte plane: The bipartite case. *Combinatorics, Probability, and Computing*, 1:181–187, 1992.

[Wel93]   D.J.A. Welsh. *Complexity: Knots, Colourings and Counting*, volume 186 of *London Mathematical Society Lecture Notes Series*. Cambridge University Press, 1993.