

EXPLAINING THE DECOMPOSITIONALITY OF MONADIC SECOND ORDER LOGIC USING APPLICATIONS TO COMBINATORICS

TOMER KOTEK

ABSTRACT. Monadic Second Order logic (MSO) is a powerful logic on graphs and graph-like structures. The usefulness of MSO comes from the decompositionality of MSO over tree-like structures. Courcelle's theorem is the classical decompositional theorem for MSO on graphs of bounded tree-width. Courcelle's theorem forms a strong theoretical basis for the development of formal methods. This extended abstract argues that a good intuition into the underlying ideas behind Courcelle's theorem can be achieved by considering the applications of these ideas to combinatorics. Technical notions regarding computation are eliminated and replaced with simple notions of recurrence relations.

Monadic Second Order logic (MSO) is the extension of First Order logic that additionally allows quantification over sets of vertices and sets of edges. MSO is very useful as a basis for development of formal methods. For instance, verification of correctness of programs with dynamically allocated memory using MSO was studied in [9], node selection from a tree in the context of XML and on information extraction from Web pages (wrapping) was studied in [8], and hardware verification based on describing circuits in MSO was studied in [1].

MSO is useful due to its high expressivity on the one hand and its decompositionality over trees and tree-like structures on the other hand. The most important decompositional theorem of MSO is the seminal result known as Courcelle's theorem [3]. Two closely related formulations of Courcelle's theorem are:

Theorem 1 (Courcelle's theorem [3], see also [4]).

- (1) *For every $k \geq 1$, The MSO theory of graphs tree-width k is decidable.*
- (2) *The model checking problem of MSO is fixed parameter tractable with respect to tree-width.*

A variation of Theorem 1 in which MSO is replaced with MSO_1 is proven in [5] and tree-width is replaced with the more general parameter clique-width; MSO_1 is the restriction of MSO in which quantification is not allowed over sets of edges, only sets of vertices.

Both formulations of Theorem 1 are based on (i) the compositionality of MSO on graphs of bounded tree-width, and (ii) the finiteness of the number of MSO -types of any fixed quantifier rank. To fully understand the statement of Theorem 1 one must be familiar with several non-trivial concepts: the syntax and semantics of MSO , tree-width, and decidability or fixed parameter tractability (the latter can be exchanged for $PTIME$ computability in a weaker formulation of Theorem 1.2).

The most problematic of these concepts are probably those related to computation and complexity. The definition of tree-width is rather complicated, but bounded tree-width can be replaced with simpler classes of graphs such as forests, cycles and series-parallel graphs. *MSO* can be approached using examples and explained in a graph-theoretic oriented approach as e.g. in [7, 11]. *MSO* can also be replaced by simpler fragments of it, or even by restricting our attention to problems which can be defined by existence of a homomorphism from G to a fixed graph H , such as existence of a 3-coloring. It is less clear how to reduce decidability or polynomial-time computability to more intuitive notions. To understand Theorem 1, one needs to understand the distinction between decidability and semi-decidability, or between polynomial-time computability and less efficient algorithms. Another problem is to explain *why* we are interested in decidability of theories to students who are not familiar with logic and its applications.

The spirit underlying Courcelle's theorem can be conveyed to curious students without elucidating these concepts by turning instead to applications of related methods in combinatorics.

Applications of logic to combinatorics have attracted attention in the literature. A classical result of Chomsky and Schützenberger [2] states that for every regular language L , the number $a(n)$ of words of length n in L satisfies a recurrence relation with constant coefficients, i.e. for every $n > r$,

$$a(n) = \sum_{i=1}^r c_i a(n-i)$$

where $c_1, \dots, c_r \in \mathbb{Z}$.

Makowsky and Fischer considered a graph-theoretic counterpart of this classic result. They defined *iteratively constructible graph sequences* and the *MSO*-definable counting functions and graph polynomials over these graphs. A k -graphs is a graphs G each of whose vertices is labeled with a unique label from $[k] = \{1, \dots, k\}$. A sequence $G_n: n \in \mathbb{N}$ of k -graphs is iteratively constructible if it is built from an initial k -graph G_0 by applying n times a operation σ which can add a fixed number of vertices to G_{n-1} and connect them to the vertices of G_{n-1} , i.e. $G_n = \sigma^n(G_0)$. σ is a finite composition of the following *basic operations*:

- $Add_i(G)$ adds a new isolated vertex with label i ;
- $\rho_{i \rightarrow j}(G)$ relabels all the vertices labeled i to j ;
- $\eta_{i,j}(G)$: adds all possible edges between vertices labeled i and vertices labeled j ;
- $\eta_{i,j}^b(G) = \eta_{i,j}(G)$ if there are at most b vertices labeled i or j and $\eta_{i,j}^b(G) = G$ otherwise;
- $\delta_{i,j}(G)$ deletes all edges between vertices labeled i and vertices labeled j .

A iteratively constructible sequences of graphs is *bounded* if no operations of the form $\eta_{i,j}$ are used in its construction. A sequence $G_n: n \in \mathbb{N}$ of graphs is a (bounded) iteratively constructible sequence if there is a sequence $H_n: n \in \mathbb{N}$, of k -graphs which is a (bounded) iteratively constructible sequence which is isomorphic to G_n ignoring the labels.

Examples of iteratively constructible graph sequences are the paths, cycles, cliques, wheels, complete bipartite graphs and prisms are given in Fig. 1.

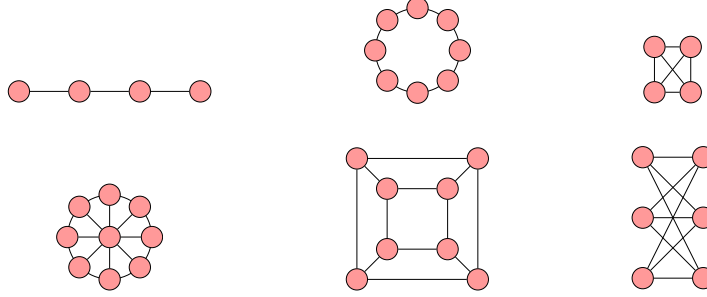


FIGURE 1. Examples of iteratively-constructible graph sequences. Of these graphs sequences, only cliques and complete bipartite graphs are not bounded.

Theorem 2 (Fischer and Makowsky 2008 [6]). *If $\mathcal{G} = \{G_n : n \in \mathbb{N}\}$ is an iteratively constructible graph sequence and $\Phi(U)$ is an MSO formula with a free second order variable such that $\Phi \in MSO_1$ or $\mathcal{G}$ is bounded, then*

$$f_{\Phi, \mathcal{G}}(n) = |\{R \subseteq V(G_n) : \langle G_n, R \rangle \models \Phi\}|$$

satisfies a linear recurrence relation

$$f_{\Phi, \mathcal{G}}(n) = \sum_{i=1}^r c_i f_{\Phi, \mathcal{G}}(n-i)$$

with constant coefficients.

As a simple consider the MSO_1 -formula $\Phi_{ind}(U) = \forall x \forall y (U(x) \wedge U(y) \rightarrow \neg E(x, y))$. The function $ind_{Paths}(n) = f_{\Phi_{ind}, Paths}(n)$ counts the number of independence sets in the path P_n . Assume w.l.o.g. that the $[n] = \{1, \dots, n\}$ is the vertex set of P_n . Let $ind_{Paths}^b(n)$, $b = 0, 1$, counts the number of independence sets R in the path P_n such that $n \notin R$ respectively $n \in R$. Then it is easy to verify that

$$\begin{aligned} ind_{Paths}(n) &= ind_{Paths}^0(n) + ind_{Paths}^1(n) \\ ind_{Paths}^0(n) &= ind_{Paths}(n-1) \\ ind_{Paths}^1(n) &= ind_{Paths}(n-2) \end{aligned}$$

This implies that $ind_{Paths}(n)$ satisfies the same recurrence as the Fibonacci numbers, and by considering that $ind_{Paths}(0) = 1$ and $ind_{Paths}(1) = 2$ we get that $ind_{Paths}(n) = F_{n+2}$. In the general proof of Theorem 2, ind_{Paths}^0 and ind_{Paths}^1 are replaced with $f_{\Phi, Paths}(n)$ for a set of formulas Φ defining the appropriate MSO-type.

One can obtain intuition regarding the decomposition property of MSO on more general graphs by considering the *bi-iteratively constructible graph sequences* introduced in [10]. Bi-iteratively constructible k -graphs sequences are built similarly to iteratively constructible graph sequences, except one of the operations building G_n from G_{n-1} is applied n times. More precisely, three operations $\sigma_1, \sigma_2, \sigma_3$ are used, $\theta_n = \sigma_1 \circ \sigma_2 \circ \dots \circ \sigma_2 \circ \sigma_3$ is their composition with σ_2 occurring n times, and $G_n = \theta_n(\theta_{n-1}(\dots(G_0)\dots))$. The operations $\sigma_1, \sigma_2, \sigma_3$ are finite compositions of basic operations. A bi-iteratively constructible sequences of graphs is *bounded* if

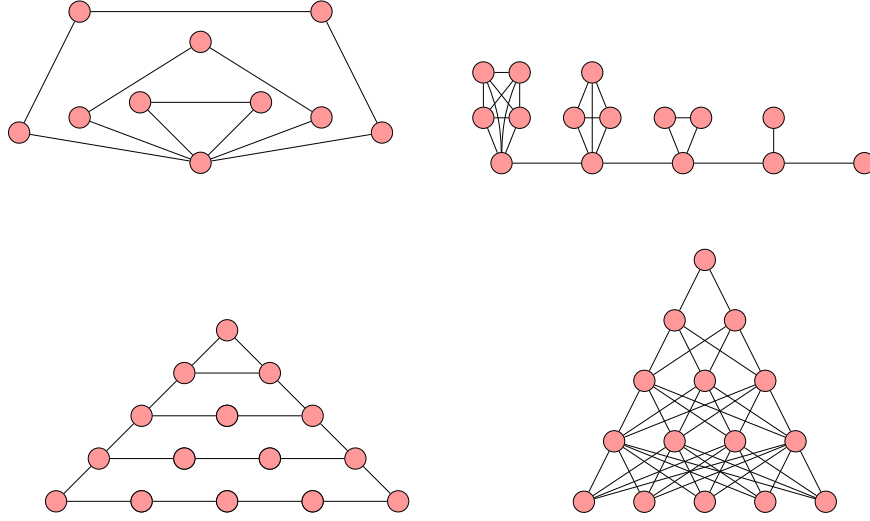


FIGURE 2. Examples of bi-iteratively constructible graphs sequences. The examples on the left are bounded and those on the right are not bounded.

no operations of the form $\eta_{i,j}$ are used in its construction. A sequence $G_n: n \in \mathbb{N}$ of graphs is a (bounded) bi-iteratively constructible sequence if there is a sequence $H_n: n \in \mathbb{N}$ of k -graphs which is a (bounded) bi-iteratively constructible sequence which is isomorphic to G_n ignoring the labels.

Some examples of bi-iteratively constructible graph sequences are given in Fig. 2.

More examples are obtained from iteratively constructible families by considering only square indices: $\mathcal{G} = \{G_n = P_{n^2} : n \in \mathbb{N}\}$ is a bi-iteratively constructible family.

Theorem 3 (Kotek and Makowsky [10]). *If $\mathcal{G} = \{G_n : n \in \mathbb{N}\}$ is a bi-iteratively constructible graph sequence and $\Phi(U)$ is an MSO formula with a free second order variable such that $\Phi \in \text{MSO}_1$ or $\mathcal{G}$ is bounded, then $f_{\Phi, \mathcal{G}}(n)$ satisfies a linear recurrence relation*

$$f_{\Phi, \mathcal{G}}(n) = \sum_{i=1}^r c_i(n) f_{\Phi, \mathcal{G}}(n-i)$$

with coefficients $c_i(n)$ which themselves satisfy linear recurrence relations with constant coefficients.

Let $\mathcal{S} = \{S_n : n \in \mathbb{N}\}$, where S_0 is the empty graph and S_n is obtained from a path P_n on $[n]$ by attaching a clique of size K_i to i for all $i \in [n]$. S_5 appears in the left bottom graph in Fig. 2. Let $\text{ind}_{\mathcal{S}}(n) = f_{\Phi_{\text{ind}}, \mathcal{S}}(n)$. Let $\text{ind}_{\mathcal{S}}^0(n)$ ($\text{ind}_{\mathcal{S}}^1(n)$) be the number of independent sets $R \subseteq V(R_n)$ such that the last vertex $n+1$ of the path belongs to R (respectively does not belong to R). Then,

$$\begin{aligned} \text{ind}_{\mathcal{S}}(n) &= \text{ind}_{\mathcal{S}}^0(n) + \text{ind}_{\mathcal{S}}^1(n) \\ \text{ind}_{\mathcal{S}}^1(n) &= \text{ind}_{\mathcal{S}}^0(n-1) \\ \text{ind}_{\mathcal{S}}^0(n) &= n \cdot \text{ind}_{\mathcal{S}}(n-1) \end{aligned}$$

$ind_S^1(n) = ind_S^0(n-1)$ reflects fact that if n belongs to the sets R counted by $ind_S^1(n)$, then $n-1$ cannot belong to R . $ind_S^0(n) = (1+n)ind_S(n-1)$ reflects that since n does not belong to the sets R counted by $ind_S^0(n)$, there are two options: either exactly one of the other $n-1$ vertices of the clique adjacent to n belongs to R , or no vertex of that clique belongs to R .

This leads to the recurrence relation

$$ind_S(n) = n \cdot ind_S(n-1) + (n-1)ind_S(n-2).$$

Theorems 2 and 3 are somewhat simplified here. For instance, counting 3-colorings is also covered by these results.

The applications of logic to graph theory and combinatorics described in this abstract are simple and appealing. They are illustrated well using simple examples. The proof of some cases rely almost only on high school background in mathematics. For instance, one such case is Theorem 2 when $f_{\Phi, \mathcal{G}}(n)$ is replaced by the number $hom_H(G_n)$ of homomorphisms from G_n to H . This case includes e.g. $ind_G(n)$ and the number of 3-colorings. Since discrete mathematics is the mathematical backbone of computer science, it is natural to teach the applications of logic to this domain.

REFERENCES

- [1] D. Basin and N. Klarlund. Automata based symbolic reasoning in hardware verification. *Formal Methods in System Design*, 13(3):255–288, 1998.
- [2] N. Chomsky and M.P. Schützenberger. The algebraic theory of context free languages. In P. Brafford and D. Hirschberg, editors, *Computer Programming and Formal Systems*, pages 118–161. North Holland, 1963.
- [3] B. Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and computation*, 85(1):12–75, 1990.
- [4] B. Courcelle and J. Engelfriet. *Graph Structure and Monadic Second-Order Logic: A Language-Theoretic Approach*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2012.
- [5] 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.
- [6] E. Fischer and J.A. Makowsky. Linear recurrence relations for graph polynomials. In *Pillars of Computer Science*, volume 4800 of *Lecture Notes in Computer Science*, pages 266–279. Springer, 2008.
- [7] B. Godlin, T. Kotek, and J.A. Makowsky. Evaluations of graph polynomials. In *Workshop on Graph Theoretic Concepts in Computer Science (WG)*, pages 183–194, 2008.
- [8] G. Gottlob and C. Koch. Monadic datalog and the expressive power of languages for web information extraction. *Journal of the ACM (JACM)*, 51(1):74–113, 2004.
- [9] J. L. Jensen, M. E. Jørgensen, M. I. Schwartzbach, and N. Klarlund. *Automatic verification of pointer programs using monadic second-order logic*, volume 32. ACM, 1997.
- [10] T. Kotek and J. A Makowsky. Recurrence relations for graph polynomials on bi-iterative families of graphs. *European Journal of Combinatorics*, 41:47–67, 2014.
- [11] L. Lovász. *Large Networks and Graph Limits*. American Mathematical Society colloquium publications. American Mathematical Society, 2012.

VIENNA UNIVERSITY OF TECHNOLOGY, kotek@forsyte.at