Monadic second-order logic for hypergraphs and matroids

Dillon Mayhew

Abstract

We consider the version of monadic second-order logic introduced in Hliněný's analogue of Courcelle's Theorem. This language establishes a natural connection between formal logics and the computational theory of hypergraphs. The theorems by Courcelle and Seese provide a framework for discussing this connection. Many of the ideas we use have their origins in the classical theorem by Büchi, Elgot, and Trakhtenbrot showing that a set of strings forms a regular language if and only if it is defined by a monadic sentence.

1 Introduction

The topics in this survey lie in the intersection of three areas: structural discrete mathematics, the theory of computation, and logic and model theory. We are motivated to work in this intersection because we can gain insight into algorithms for graphs and other discrete structures by considering the power of certain logical languages. This connection has produced a great deal of beautiful mathematics, as exemplified by the theorems of Courcelle [13] and Seese [52]. The first of these is a meta-theorem providing polynomial-time recognition algorithms for graph properties defined by sentences in a particular logical language (as long as the input class has bounded structural complexity). Seese's Theorem describes when a minor-closed class of graphs has a decidable theory (meaning there is a finite procedure for deciding whether or not any given sentence in the logical language holds for every graph in the class).

The theorems by both Courcelle and Seese are focused on *monadic second-order logic*. This is a fragment of full second-order logic in which we can quantify over elements and subsets of the domain, but not over any relation that is binary, ternary, or of any higher arity. Both theorems apply to monadic second-order languages for graphs. This survey will explore less-familiar territory: the monadic second-order logic of hypergraphs, especially matroids. Although the monadic logic of discrete objects has been well explored (for example in the monograph by Courcelle and Engelfriet [15]), a different flavour emerges when we consider hypergraphs, as we shall discuss.

Let us start with a refresher of some fundamental definitions. A *hypergraph* is a pair $(E, \mathcal{I})$, where E is a finite set and $\mathcal{I}$ is a family of subsets of E. Quite often, E is called the vertex-set of the hypergraph. In matroidal contexts, E is frequently the set of edges of some other graph. Referring to these edges as vertices would be disastrously confusing, so we adopt the convention of calling E the *ground set* of the hypergraph. The members of $\mathcal{I}$ are the *hyperedges*.

Matroids are axiomatic abstractions of various notions of independence, including linear independence. We will defer the formal definition until Section 2.2. For now, it suffices to note that a matroid is a hypergraph satisfying additional axioms. Frequently, we think of the hyperedges as being *independent sets*. This means the

family of hyperedges will satisfy axioms that capture properties of linear independence. However, there is a famously large number of ways of viewing a matroid as a hypergraph. Just as a topology may be viewed as a collection of closed sets, open sets, or bases, a matroid may be considered to be a collection of independent sets, bases, or circuits, amongst many other possibilities.

Let us consider how we might develop a logical language that would allow us to make statements about matroids or other hypergraphs. We will present three possibilities, and argue that one is a little too weak, another a little too strong, and the third is about right. The first approach uses *first-order logic*. In this model, we have a supply of variables $z_1, z_2, z_3, \ldots$ which will be interpreted as elements of the ground set. We will allow ourselves a binary equality predicate, and for each non-negative integer k, we have a k-arity predicate I_k, which is interpreted as a subset of k-tuples of the ground set. In this approach, the k-element independent sets of the matroid are exactly the sets that can be ordered into a k-tuple that satisfies the I_k predicate. Since independent sets are unordered, the axiom schema for matroids will need to assert that any reordering of an independent subset will produce another independent subset. For example, we will need an axiom such as

$$I_3(z_1, z_2, z_3) \to I_3(z_1, z_3, z_2) \wedge I_3(z_2, z_1, z_3) \wedge I_3(z_2, z_3, z_1) \wedge I_3(z_3, z_1, z_2) \wedge I_3(z_3, z_2, z_1)$$

or something equivalent.

This first-order approach is the one used by Vámos, in his beautifully-titled paper *The missing axiom of matroid theory is lost forever* [59]. The advantage of the first-order system is that we have access to the tools of classical model theory, including compactness, which Vámos uses in a fundamental way. But the language is too weak for many natural purposes. For example, first-order logic is not able to say that a matroid has a given minor. Minors are central to the structural theory of matroids, so this is one reason we reject first-order logic as being too weak.

We move on to another approach. The adjective *monadic* means that we are allowed to quantify only over variables and unary relations. Quantifying over a unary relation is the same as quantifying over a subset of the domain. In full second-order logic, we would be able to quantify over all possible relations, of any possible arity. In the 2-*sorted* monadic second-order logic of hypergraphs, we have one sort of variables $z_1, z_2, z_3, \ldots$ representing elements of the ground set and another sort $h_1, h_2, h_3, \ldots$ representing the hyperedges. We also have variables $Z_1, Z_2, Z_3, \ldots$ representing subsets of elements, and $H_1, H_2, H_3, \ldots$ representing subsets of hyperedges. We let ourselves quantify over all these variables. We give ourselves an equality predicate as well as binary predicates of the form $z_i \in Z_j$, $h_i \in H_j$, and $z_i \in h_j$, which are interpreted exactly as one would expect. We denote this logic by $\mathrm{CMSO}_2^{\mathrm{hyp}}$.

I suggest that $\mathrm{CMSO}_2^{\mathrm{hyp}}$ is stronger than we would like. In particular, it is strong enough that we should not expect an analogue of Courcelle's Theorem to hold. To be specific, $\mathrm{CMSO}_2^{\mathrm{hyp}}$ can make the statement that the ground set can be partitioned into hyperedges:

$$\exists H_1 \ (\forall z_1 \ \exists h_1 \ (z_1 \in h_1 \wedge h_1 \in H_1 \wedge \forall h_2 \ (z_1 \in h_2 \wedge h_2 \in H_1 \to h_1 = h_2))).$$

The problem of deciding whether a hypergraph contains a partition into hyperedges is the EXACT COVER problem, one of Karp's original 21 NP-complete problems [36]. Therefore it is as computationally difficult as any problem that can be

solved by a polynomial-time non-deterministic Turing machine. Consequently we do not expect it can be solved by a (deterministic) polynomial-time algorithm. In the classical version of EXACT COVER, the hypergraph is described by a list of its hyperedges. I speculate that it remains NP-complete even when the hypergraph is described via a tree automaton with a bounded number of states (Conjecture 7.1). This would imply that an analogue of Courcelle's Theorem is unlikely to hold for hypergraphs using the logic $\mathrm{CMSO}_2^{\mathrm{hyp}}$. So $\mathrm{CMSO}_2^{\mathrm{hyp}}$ is strong enough that we seem to lose some of the prime motivation for studying monadic second-order logic: the correspondence between the logical language and the theory of computation.

For this reason, we are going to concentrate on a third approach: the "Goldilocks" logic for hypergraphs, which we denote by $\mathrm{CMSO}_1^{\mathrm{hyp}}$. This is the logic introduced by Petr Hliněný when proving a version of Courcelle's Theorem for representable matroids (Theorem 3.2). In $\mathrm{CMSO}_1^{\mathrm{hyp}}$ we have variables $Z_1, Z_2, Z_3, \ldots$ which are interpreted as subsets of the ground set. We are able to quantify over these variables. We allow ourselves a binary predicate $\subseteq$ to let us speak about the relation between subsets, and we have a unary predicate $\mathsf{hyp}(\cdot)$, which will be satisfied exactly when the input variable is interpreted as a hyperedge. Note that in $\mathrm{CMSO}_1^{\mathrm{hyp}}$ we have dropped variables representing elements of the ground set and we use only subset variables. We lose no expressive power with this change, because it is easy to state when a variable represents the empty subset or a singleton subset.

If a class of hypergraphs has bounded structural complexity (as defined in Section 2.6), then any subclass that is defined in $\mathrm{CMSO}_1^{\mathrm{hyp}}$ can be recognised by a tree automaton (Lemma 2.5). This means that analogues of Courcelle's Theorem do hold in this new regime, as we discuss in Section 3. The major difference between $\mathrm{CMSO}_1^{\mathrm{hyp}}$ and more classical variants of monadic second-order logic such as $\mathrm{CMSO}_2^{\mathrm{hyp}}$ is the presence of predicates that do not have fixed arity. In most cases the *signature* of a monadic second-order logic will specify, amongst other things, the relational predicates used in the language. Importantly, each such predicate is interpreted as a set of tuples of the ground set, and each predicate has a specified *arity*, which is simply the number of elements in those tuples. In $\mathrm{CMSO}_1^{\mathrm{hyp}}$, we do not want predicates operating on a fixed number of elements of the ground set. That would require us to use an infinite sequence of predicates to talk about hyperedges of unbounded size, just as in the first-order logic of matroids. So instead the predicate $\mathsf{hyp}(\cdot)$ operates on variables that represent subsets of arbitrary size. (Quite often, traditional monadic second-logic allows predicates operating on subset variables when we want to speak of the cardinality of a subset modulo some integer. The presence of these predicates distinguishes monadic second-order logic and *counting monadic second-order logic*. So in some ways $\mathrm{CMSO}_1^{\mathrm{hyp}}$ is just extending this idea by introducing another such predicate.) Having predicates that operate on subset variables gives the monadic second-order logic of hypergraphs and matroids a different flavour to the fixed-arity logic used in the context of graphs. We shall talk more about this difference in Section 4 while discussing Seese's Conjecture.

1.1 The Büchi-Elgot-Trakhtenbrot Theorem

We are exploring the interaction between logical languages and questions from complexity theory and computability theory. Many of the important ideas in this

interaction can be found in a classical theorem which was proved independently by Büchi [11], Elgot [18], and Trakhtenbrot [56] (the BET Theorem). By analysing this theorem, we give ourselves a framework for understanding much of the work in the survey. To state the BET Theorem we need to discuss strings over finite alphabets and a formal language which can express statements about those strings. For any positive integer n, we use $[n]$ to denote $\{1, 2, \ldots, n\}$. If n is zero, then $[n]$ is the empty set. From this point onwards, Σ stands for a finite set called an *alphabet*.

Definition 1.1 (Strings and languages) A (finite) *string* (over Σ) is a function $\mathbf{w}\colon [n] \to \Sigma$, for some non-negative integer n. The elements of $[n]$ are called the *positions* of the string. The set of strings over Σ is denoted by Σ^*. Any subset of Σ^* is a *language*.

Next we develop a version of monadic second-order logic for strings. The variables we use are $Z_1, Z_2, Z_3, \ldots$, which are interpreted as subsets of positions. The *atomic formulas* include $Z_i \subseteq Z_j$ which is interpreted in the obvious way, as well as:

- $\mathsf{term}(Z_i)$, which is interpreted as saying that Z_i represents a *terminal* set of positions, meaning a set of the form $\{i, i+1, \ldots, n\}$ for a word $\mathbf{w}\colon [n] \to \Sigma$.
- $\mathsf{char}_\alpha(Z_i)$, where α is an element of the alphabet Σ. This predicate is interpreted as saying that Z_i contains a single position and the character in this position is equal to α.

Now we consider the collection of formulas that we can construct using the atomic formulas, the usual logical connectives $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$, and the quantifiers $\forall$ and $\exists$. This collection is denoted by MSO^Σ. A variable is *bound* in a formula if it is quantified over, and otherwise it is *free*. A formula with no free variables is a *sentence*. We write $Z_i = Z_j$ to mean $(Z_i \subseteq Z_j) \wedge (Z_j \subseteq Z_i)$ and $Z_i \nsubseteq Z_j$ to stand for $\neg(Z_i \subseteq Z_j)$. Let $\mathsf{empty}(Z_i)$ be the formula $\forall Z_j\ (Z_j \subseteq Z_i \rightarrow Z_j = Z_i)$. Note that $\mathsf{empty}(Z_i)$ is satisfied exactly when Z_i is interpreted as the empty set. Next let $\mathsf{sing}(Z_i)$ be $\neg\mathsf{empty}(Z_i) \wedge \forall Z_j\ (Z_j \subseteq Z_i \rightarrow (\mathsf{empty}(Z_j) \vee Z_j = Z_i))$. Thus $\mathsf{sing}(Z_i)$ is satisfied exactly when Z_i is interpreted as a singleton set.

Let $\mathcal{L} \subseteq \Sigma^*$ be a language. We say that $\mathcal{L}$ is MSO^Σ*-definable* if there is some sentence that is satisfied by exactly the strings in $\mathcal{L}$.

Example 1.2 Let Σ be the alphabet $\{A, B\}$. We consider the language defined by the MSO^Σ-sentence $\exists Z_1\ (\mathsf{term}(Z_1) \wedge \forall Z_2\ (\mathsf{sing}(Z_2) \rightarrow (\mathsf{char}_B(Z_2) \leftrightarrow Z_2 \subseteq Z_1)))$. This language is the subset of Σ^* consisting of all strings where no A symbol comes after a B symbol.

Definition 1.3 (Finite-state automata) Let Σ be an alphabet. A finite-state automaton (FSA) (with alphabet Σ) is a tuple $(D, \mathsf{start}, \mathsf{acc})$, where D is a directed graph, start is a vertex of D, and acc is a set of vertices of D. Each arc of D is labelled with a single element of Σ. The vertices of D are known as *states*, and we use Q to denote the set of states. Generally, "state" refers to a possible configuration of the memory in a computing machine. We call start the *initial state* and refer to acc as the set of *accepting states*. Note that we allow self-directed loops and parallel arcs.

Say that $\mathbf{D} = (D, \mathsf{start}, \mathsf{acc})$ is an FSA and that $\mathbf{w}\colon [n] \to \Sigma$ is a string. A *run* of $\mathbf{D}$ on $\mathbf{w}$ is a function $r\colon \{0\} \cup [n] \to Q$ such that $r(0) = \mathsf{start}$ and for every $i \in [n]$, if $\mathbf{w}(i) = \alpha$, then there is an arc from $r(i-1)$ to $r(i)$ that is labelled with α. If $r(n)$ is in acc, then the run *accepts* $\mathbf{w}$. We say that $\mathbf{D}$ *accepts* $\mathbf{w}$ if there exists an accepting run.

Let $\mathcal{L} \subseteq \Sigma^*$ be a language. We say that $\mathcal{L}$ is *regular* if there exists an FSA that accepts exactly the strings in $\mathcal{L}$. In this case we say that the FSA *recognises* the language.

Example 1.4 Let Σ be the alphabet $\{A, B\}$. Let D be the directed graph shown in Figure 1. Let acc be the set of shaded vertices. A string is accepted by $(D, \mathsf{start}, \mathsf{acc})$ if and only if there is no A character that appears after a B character.

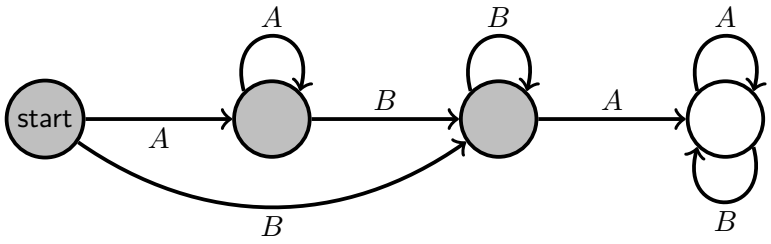

Figure 1: A finite-state automaton

It is no coincidence that the language defined in Example 1.2 is the same as the language recognised by the FSA in Example 1.4. The Büchi-Elgot-Trakhtenbrot Theorem tells us we are always able to construct these parallel definitions of regular languages. In fact the languages defined by the monadic language MSO^Σ are exactly the languages recognised by finite-state automata.

Theorem 1.5 (BET Theorem) *Let Σ be a finite alphabet and let $\mathcal{L} \subseteq \Sigma^*$ be a language. Then $\mathcal{L}$ is regular if and only if it is MSO^Σ-definable.*

The proof of the BET Theorem contains many of the seminal ideas linking the monadic languages of discrete structures to the theory of computing machines. We will spend some time sketching the proof, since the key ideas inform so much of the work in this survey. The easier direction involves proving that a regular language is MSO^Σ-definable. We assume that $\mathbf{D} = (D, \mathsf{start}, \mathsf{acc})$ is an FSA that recognises the language $\mathcal{L}$. We construct an MSO^Σ-sentence $\varphi_{\mathbf{D}}$ that defines $\mathcal{L}$. The idea is that $\varphi_{\mathbf{D}}$ will assert the existence of an accepting run of $\mathbf{D}$ on a string. Let $Q = \{q_1, q_2, \ldots, q_s\}$ be the set of states of D. We start the construction by asserting that there is a partition $\{P_1, P_2, \ldots, P_s\}$ of the set of positions. We are essentially labelling each position in the string with one of the states $q_1, q_2, \ldots, q_s$, and this is equivalent to creating a function $r\colon \{0\} \cup [n] \to Q$ for any string with n positions. The job of $\varphi_{\mathbf{D}}$ is to enforce that this function is an accepting run of $\mathbf{D}$ on the string. We must assert that r takes the final position to an accepting state, and $\varphi_{\mathbf{D}}$ does this by saying that any terminal singleton set belongs to $\cup_{q_i \in \mathsf{acc}} P_i$. Also, φ needs to ensure that every transition from one state to another obeys the arc-labellings

of D. Imagine that the arcs of D that depart from q_a with the label $\alpha \in \Sigma$ have $q_{k_1}, q_{k_2}, \ldots, q_{k_s}$ as their destinations. Then $\varphi_{\mathbf{D}}$ will have a clause saying that if Z_i and Z_j are singleton sets of positions and there is exactly one terminal set that contains Z_j but not Z_i, and $\mathsf{char}_\alpha(Z_i)$ holds, and Z_i is in P_a, then Z_j is in one of $P_{k_1}, P_{k_2}, \ldots, P_{k_s}$. We repeat this construction for every choice of state q_a and label α. We also need to assert that the state assigned to the first position is appropriate. We do this with a clause saying that if the singleton set Z_i is in exactly one terminal set and $\mathsf{char}_\alpha(Z_i)$ holds, then Z_i is in one of $P_{k_1}, P_{k_2}, \ldots, P_{k_s}$, where $q_{k_1}, q_{k_2}, \ldots, q_{k_s}$ are the destinations of arcs departing start with the label α. With this, we have sketched the construction of $\varphi_{\mathbf{D}}$. It is not difficult to check that a string satisfies $\varphi_{\mathbf{D}}$ if and only if it is accepted by $\mathbf{D}$. So we have completed the "only if" direction of the BET Theorem.

The "if" direction is more convoluted, since it involves using the power of non-determinism to simulate the existential quantifiers in an MSO^Σ-sentence. Let $\mathbf{D} = (D, \mathsf{start}, \mathsf{acc})$ be an FSA. Then $\mathbf{D}$ is *deterministic* if, for every state q and every label $\alpha \in \Sigma$, there is exactly one arc departing from q that is labelled with α. Otherwise the automaton is *non-deterministic*. If the automaton is deterministic, then there is exactly one run of the automaton on any string. In a non-deterministic FSA, there may be multiple runs (or zero runs) on a given string. Given that the power of non-determinism vs. determinism in general computing machines leads to the notorious P vs. NP problem, it is still a little surprising that non-determinism gives finite-state automata no additional power, as was demonstrated by Rabin and Scott in 1959 [48].

Lemma 1.6 *Let $\mathbf{D} = (D, \mathsf{start}, \mathsf{acc})$ be an FSA. There is a deterministic FSA that accepts exactly the same strings as $\mathbf{D}$.*

Sketch proof. We replace the non-deterministic machine $\mathbf{D}$ with a deterministic automaton $\mathbf{D}_{\mathrm{det}}$. The price we must pay is an exponential blow-up in the number of states. If Q is the set of states of D, then $\mathbf{D}_{\mathrm{det}}$ will have the power set of Q as its set of states. When $A, B \subseteq Q$ are sets of states, there is an arc labelled with $\alpha \in \Sigma$ from A to B in $\mathbf{D}_{\mathrm{det}}$ if and only if B is the set of states q such that there exists $q' \in A$ and an arc of D from q' to q that is labelled by α. That is, the unique arc departing from A labelled by α has as its destination the set of states that can be reached from a state in A using an arc labelled by α. This makes it clear that $\mathbf{D}_{\mathrm{det}}$ is indeed deterministic. The starting state of $\mathbf{D}_{\mathrm{det}}$ is $\{\mathsf{start}\}$, and the accepting states are the subsets of Q that have non-empty intersection with acc. It is very easy to see that if there is an accepting run of $\mathbf{D}$ on $\mathbf{w} \in \Sigma^*$, then there is an accepting run of $\mathbf{D}_{\mathrm{det}}$. There is a fairly mechanical inductive proof which shows that an accepting run of $\mathbf{D}_{\mathrm{det}}$ implies the existence of an accepting run of $\mathbf{D}$. Thus $\mathbf{D}$ and $\mathbf{D}_{\mathrm{det}}$ accept exactly the same strings. □

To complete the proof of the BET Theorem, we must show that any language defined by a sentence φ in MSO^Σ can be recognised by an automaton. As is often the case in model theory, the proof is inductive on the number of steps needed to construct the sentence φ. The issue with this strategy is that the process of building a sentence involves formulas that feature free variables, and which therefore are not sentences. So our inductive strategy requires us to prove that a language defined by

any formula (possibly with free variables) can be recognised by an automaton. The problem is that it makes no sense to ask whether a formula with free variables is satisfied by a string, just as it makes no sense to ask whether the equation $x = y$ is correct without specifying what these variables stand for. So we introduce a function which will give meaning to the free variables by assigning values to them. Let φ be some formula in MSO^{Σ} and let F be its set of free variables. Let $\mathbf{w}\colon [n] \to \Sigma$ be a string. An *interpretation* is a function $\theta\colon F \to 2^{[n]}$. Now we say that φ is satisfied by the pair $(\mathbf{w}, \theta)$ if it is true when every variable in F is interpreted as its image under θ.

Having specified what it means for a formula to be satisfied, we must explain how an automaton can process a string along with an interpretation of free variables. We do this by stacking the string on top of a sequence of strings from $\{\checkmark, \boldsymbol{\times}\}^*$. Each of these additional strings defines a subset of $[n]$ — the subset of positions that contain a $\checkmark$ symbol. We now make this more formal. As always, Σ is a finite alphabet. Assume that n is a non-negative integer and that $\mathbf{w}\colon [n] \to \Sigma$ is a string. Let s be a non-negative integer and let $P_1, P_2, \ldots, P_s$ be subsets of $[n]$. We define the string

$$S(\mathbf{w}; P_1, P_2, \ldots, P_s)\colon [n] \to \Sigma \times \{\checkmark, \boldsymbol{\times}\}^s.$$

For any position $i \in [n]$, the image of i under $S(\mathbf{w}; P_1, P_2, \ldots, P_s)$ is the tuple $(\mathbf{w}(i), P_1(i), P_2(i), \ldots, P_s(i))$, where $P_j(i)$ is $\checkmark$ if i is in P_j, and $\boldsymbol{\times}$ otherwise. A finite-state automaton with alphabet $\Sigma \times \{\checkmark, \boldsymbol{\times}\}^s$ is said to be a Σ-*FSA* with *arity* s. The next result provides the final step in the proof of the BET Theorem. It uses a "boot-strapping" approach that starts with the atomic formulas and builds automata that can recognise any arbitrary formula.

Lemma 1.7 *Let Σ be a finite alphabet. Let φ be a formula in MSO^{Σ} and let $Z_{i_1}, Z_{i_2}, \ldots, Z_{i_s}$ be the free variables of φ. There exists $\mathbf{D}$, a Σ-FSA with arity s, where the following property is satisfied: for any string $\mathbf{w}\colon [n] \to \Sigma$ and any choice of subsets $P_1, P_2, \ldots, P_s \subseteq [n]$, we have that $\mathbf{D}$ accepts $S(\mathbf{w}; P_1, P_2, \ldots, P_s)$ if and only if φ is satisfied by $(\mathbf{w}, \theta)$, where θ is the interpretation that takes each Z_{i_j} to P_j.*

Sketch proof. We again apply induction on the number of steps required to build φ from atomic formulas. This means there are three base cases, corresponding to the three possible forms of atomic formula. In these three cases, φ is one of the formulas $Z_{i_1} \subseteq Z_{i_2}$, or $\mathsf{term}(Z_{i_1})$, or $\mathsf{char}_{\alpha}(Z_{i_1})$. In the first case, $\mathbf{D}$ will be an FSA with alphabet $\Sigma \times \{\checkmark, \boldsymbol{\times}\}^2$. We need $\mathbf{D}$ to accept if and only if it never encounters a triple of the form $(\alpha, \boldsymbol{\times}, \checkmark)$, because this will correspond to an element of $[n]$ that is in P_2 but not in P_1. We leave it as an exercise to construct $\mathbf{D}$ in the other cases.

For the inductive step of the argument, we make our lives simpler by noting that any formula is logically equivalent to one constructed using only negation $\neg$, conjunction $\wedge$, and the existential quantifier $\exists$. This leaves us with three cases to which we apply the inductive strategy. To start with, we assume that φ is a formula of the form $\neg\psi$. Note that the free variables of φ are exactly the free variables of ψ. Because ψ is constructed using fewer steps than $\varphi = \neg\psi$, the inductive hypothesis implies there exists a Σ-FSA $\mathbf{D}'$ that accepts $S(\mathbf{w}; P_1, P_2, \ldots, P_s)$ if and only if ψ is satisfied by $\mathbf{w}$ and the interpretation θ. Lemma 1.6 shows that we may as well

assume $\mathbf{D}'$ is deterministic. Now the rest of this case is easy: we construct $\mathbf{D}$ from $\mathbf{D}'$ by making the accepting states non-accepting, and vice versa. This means that $\mathbf{D}$ will accept precisely when $\mathbf{D}'$ does not accept, which is exactly the behaviour that we want. Note that this argument relies on the determinism of $\mathbf{D}'$ in a critical way, because we are assuming there is a unique state that $\mathbf{D}'$ can arrive at as it finishes its computation on $S(\mathbf{w}; P_1, P_2, \ldots, P_s)$.

If φ is equal to $\psi_1 \wedge \psi_2$, then we let $\mathbf{D}_1$ and $\mathbf{D}_2$ be the automata provided to us by the inductive hypothesis, where $\mathbf{D}_i$ decides the formula ψ_i. There are some technicalities in this case, since a free variable of ψ_1 may not appear in ψ_2. Thus the arities of $\mathbf{D}_1$ and $\mathbf{D}_2$ may be different. But we will elide these difficulties, since the idea of the construction is fairly simple: we construct the FSA $\mathbf{D}$ as the product of $\mathbf{D}_1$ and $\mathbf{D}_2$. By this we mean that the set of states in $\mathbf{D}$ is the Cartesian product of the set of states of $\mathbf{D}_1$ and $\mathbf{D}_2$. The starting state of $\mathbf{D}$ is the ordered pair containing the two starting states of $\mathbf{D}_1$ and $\mathbf{D}_2$. A ordered pair of states in $\mathbf{D}$ is accepting in $\mathbf{D}$ if and only if the two states are accepting in $\mathbf{D}_1$ and $\mathbf{D}_2$. There is an arc from (q_1, q_2) to (q_1', q_2') with the label α if and only if each $\mathbf{D}_i$ has an arc labelled by α from q_i to q_i'.

The final case of the proof is where φ is the formula $\exists Z_i \psi$, where Z_i is a free variable in ψ. This is the case where we use non-determinism to mimic the power of the existential quantifier. By induction, there is an FSA $\mathbf{D}'$ that recognises the formula ψ. Say that the arity of $\mathbf{D}'$ is $s+1$. Thus every arc-label of $\mathbf{D}'$ has the form $(\alpha, c_1, c_2, \ldots, c_{s+1})$, where each c_j belongs to $\{\checkmark, \boldsymbol{\times}\}$. For the sake of simplicity, we assume without any loss of generality that Z_i is the last of the free variables in ψ, so that it corresponds to the character c_{s+1}. The states of the FSA $\mathbf{D}$ are the same as those of $\mathbf{D}'$. The arc-labels of $\mathbf{D}$ are exactly the tuples we obtain by deleting the final character of the arc-labels in $\mathbf{D}'$. Thus the arity of $\mathbf{D}$ is one less than that of $\mathbf{D}'$, as we would expect. Let q and q' be states in $\mathbf{D}'$. There is an arc labelled $(\alpha, c_1, c_2, \ldots, c_s)$ from q to q' in $\mathbf{D}$ if and only if $\mathbf{D}'$ has an arc from q to q' labelled either $(\alpha, c_1, c_2, \ldots, c_s, \checkmark)$ or $(\alpha, c_1, c_2, \ldots, c_s, \boldsymbol{\times})$. Note that $\mathbf{D}$ will typically not be deterministic, even though $\mathbf{D}'$ may be. A simple inductive argument shows that $\mathbf{D}$ accepts $S(\mathbf{w}, P_1, P_2, \ldots, P_s)$ if and only if there is some subset P_{s+1} of positions such that $\mathbf{D}'$ accepts $S(\mathbf{w}, P_1, P_2, \ldots, P_{s+1})$, and this is exactly the behaviour we were seeking. □

We obtain the proof of the BET Theorem from Lemma 1.7 by restricting to the case where φ is a sentence (which corresponds to the case with $s = 0$). In this case, the lemma provides an FSA $\mathbf{D}$ with alphabet Σ which accepts $S(\mathbf{w}) = \mathbf{w}$ if and only if φ is satisfied by $\mathbf{w}$. This shows that the language defined by φ is recognisable by an FSA, and completes the proof of the "if" direction.

The BET Theorem is relevant to this survey for multiple reasons. The proof of the theorem provides a framework that helps us understand the proof of other related theorems, including Courcelle's Theorem and its matroid-theoretical descendants, as we see in Section 3. Let us think about the computational difficulty of recognising whether a given string belongs to a specified language. This problem can be arbitrarily hard and may not be solvable by any Turing machine. But once we know that a language is defined by a monadic sentence, then the complexity of testing membership in the language becomes almost trivial. The BET Theorem tells

us that any such language is exactly the language recognised by some FSA, and the whole point of an FSA is that it never backtracks. It therefore runs in time that is linearly bounded by the length of the string. So if a language is monadically-defined, then there is an algorithm which will test whether an input string belongs to the language and this algorithm runs in linear time. This phenomenon underlies all Courcelle-type theorems. Courcelle shows that any class of graphs with bounded structural complexity that is defined by a monadic sentence can be recognised by an automaton. This leads to an efficient algorithm for testing membership in the class. The main difficulty in upgrading the BET Theorem to Courcelle's Theorem is that graphs do not come equipped with a linear ordering in the same way that strings do. Instead we consider classes of graphs with tree-like structure (that is, classes with bounded *tree-width*), which can therefore be processed by automata that operate on trees, rather than strings.

The BET Theorem also illustrates a theme that has dominated recent research into monadic languages and discrete structures: recognisability vs. definability. The first of these terms refers to the power of a computing machine to test membership of a class and the second refers to the expressive power of a logical language to define a class. The BET Theorem shows that in at least one context, these two properties are equivalent: a language of strings is definable by a monadic sentence if and only if it is recognisable by an automaton. In other contexts, it is much more difficult to show that recognisability is equivalent to definability. It is not too hard to show that when a class of graphs with bounded tree-width is definable by a monadic sentence, it is recognisable by a tree automaton. The other direction is the subject of a conjecture by Courcelle [14, Conjecture 1] which was open for decades until it was proved in a beautiful paper by Bojańczyk and Pilipczuk [7].

Considering languages of strings can also inform our thinking around decidable theories. To show that a language of strings has a decidable monadic second-order theory, we must show that there is a computing machine which can take any sentence from MSO^{Σ} as input, and decide in finite time whether or not every string in the language satisfies the sentence. Clearly, this is equivalent to deciding if there is a string in the language that is a counterexample to the sentence. The crucial idea here is that given any finite-state automaton, there is a finite procedure for deciding if there is a string that will be accepted by the automaton. This follows from the *pumping lemma* [17, Proposition 5.3.1], which provides an upper bound on the length of a smallest-possible string that is accepted by the automaton. Say that the language $\mathcal{L}$ is regular, so that it is recognised by an automaton. The BET Theorem says there is a sentence $\varphi_{\mathcal{L}}$ that is satisfied exactly by the strings in $\mathcal{L}$. Now assume we want to test whether the arbitrary monadic sentence ψ is satisfied by all the strings in $\mathcal{L}$. We construct the automaton that accepts exactly the strings satisfying $\varphi_{\mathcal{L}} \wedge (\neg\psi)$, using the boot-strapping techniques of Lemma 1.7. The pumping lemma provides us with an explicit upper bound on the length of a smallest-possible string satisfied by this automaton. All we need do is test the strings up to that length. If any of these strings is accepted then we have found a string satisfying $\varphi_{\mathcal{L}} \wedge (\neg\psi)$, which therefore belongs to $\mathcal{L}$ but does not satisfy ψ. If none of the strings is accepted, then ψ has no counterexample in $\mathcal{L}$, so it is a theorem for this class.

We will revisit all these uses of the BET theorem throughout the survey.

2 Fundamentals

In this section we review some fundamental definitions and ideas.

2.1 Graphs and hypergraphs

A *graph*, G, consists of a set of *vertices* (denoted by $V(G)$), a set of *edges* (denoted by $E(G)$), and an *incidence function*, which takes each edge to a set containing either one or two vertices. Note that this definition allows graphs to contain *loops* (edges that are incident with a single vertex) and *parallel edges* (pairs of distinct non-loop edges which are incident with the same vertices). If X is a subset of $E(G)$, then $G[X]$ is a graph with X as its edge-set. The vertices of $G[X]$ are the vertices of G that are incident with at least one edge in X.

Recall from the introduction that a *hypergraph* (also called a *set-system*) is a pair $(E, \mathcal{I})$ where E is a finite set and $\mathcal{I}$ is a family of subsets of E. We refer to E as the *ground set* of the hypergraph and to the sets in $\mathcal{I}$ as *hyperedges*.

2.2 Matroids

Matroids are axiomatic abstractions of a notion of (in)dependence that arises naturally in linear algebra, graph theory, and the theory of field extensions, amongst other places. Birkhoff [2, 3], Mac Lane [40], Steinitz [54], and Van der Waerden [60] were among the prominent mathematicians who knew that these disparate areas of mathematics exhibited similar behaviour. The neglected mathematician Takeo Nakasawa gave a set of axioms to capture this similarity [44], but it was Hassler Whitney who has become known as the founder of matroid theory. The axioms below are essentially the same as his [61]. Any facts stated here without proof can be found in Oxley's *Matroid Theory* [47], which is the standard text for a thorough introduction to the subject.

Definition 2.1 A *matroid* is a hypergraph $(E, \mathcal{I})$ satisfying the following axioms.

(**I1**) $\mathcal{I}$ is non-empty,

(**I2**) every subset of a hyperedge is a hyperedge,

(**I3**) if I and J are hyperedges and $|I| < |J|$, then there exists an element $e \in J \backslash I$ such that $I \cup \{e\}$ is a hyperedge.

The set E is known as the *ground set* of the matroid, and the members of $\mathcal{I}$ are the *independent sets*.

A subset of the ground set that is not independent is *dependent*. A *circuit* is a dependent set that has no dependent proper subset. A *basis* is an independent set that has no independent proper superset. Any set that contains a basis is *spanning*. A set that is disjoint with at least one basis is *coindependent*. The bases of a matroid are equicardinal, and the shared cardinality is the *rank* of that matroid.

Assume that $M = (E, \mathcal{I})$ is a matroid. The coindependent sets of M are themselves the independent sets of a matroid with E as its ground set. This matroid is the *dual* of M, denoted by M^*. If X is a subset of E, then $r_M(X)$ stands for the

rank of X in M, which is the maximum cardinality of an independent subset of X. Thus the rank of M is $r_M(E)$, which we abbreviate to $r(M)$.

Let X and Y be disjoint subsets of E such that X is independent in M. Then

$$M/X\backslash Y = (E\backslash(X \cup Y), \{I \subseteq E\backslash(X \cup Y) \colon I \cup X \in \mathcal{I}\})$$

is a matroid, and any matroid arising from M in this way is a *minor* of M. We say $M/X\backslash Y$ is obtained by *contracting* X and *deleting* Y. A minor of M that is not equal to M is a *proper minor*.

If $\mathcal{M}$ is a class of matroids and any minor of a matroid in $\mathcal{M}$ is also in $\mathcal{M}$, then $\mathcal{M}$ is *minor-closed*. If $\mathcal{M}$ is a minor-closed class of matroids, then an *excluded minor* for $\mathcal{M}$ is a matroid N such that N does not belong to $\mathcal{M}$, but any proper minor of N belongs to $\mathcal{M}$. If M is a matroid, then M is in $\mathcal{M}$ if and only if it does not have a minor isomorphic to an excluded minor for $\mathcal{M}$. An *antichain* is a set $\mathcal{A}$ of pairwise non-isomorphic matroids such that whenever M and N are distinct members of $\mathcal{A}$, no minor of M is isomorphic to N. The excluded minors for a minor-closed class form an antichain. A class of matroids is *well-quasi-ordered* if it contains no infinite antichain.

Matroids can be seen as abstractions of matrices (explaining their name). To see why this is the case, let A be a matrix with entries over the field $\mathbb{F}$. If A has r rows, then every column is a vector from $\mathbb{F}^r$. We let E be the set of these vectors and we let $\mathcal{I}$ be the family of linearly independent subsets of E. Then $(E, \mathcal{I})$ is a matroid, and any such matroid is $\mathbb{F}$-*representable*.

Figure 2 shows the *Fano matroid*, which has rank 3. (You may recognise it as the projective plane PG$(2, 2)$.) The ground set of the Fano matroid is the set of points in the diagram. Because the matroid has rank 3, any subset of size four or more is dependent. All subsets of size two or less are independent. A set of size three is independent if and only if it is not contained in one of the lines of the diagram. In this diagram, the Fano matroid is represented over $\mathbb{F}_2$, the field of two elements, so each point is labelled with a vector from $\mathbb{F}_2^3$. We see that when a matroid is representable it means that its points can be given coordinates in a way that captures the dependencies of the matroid.

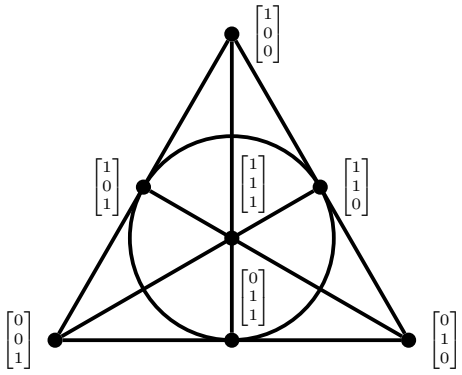

Figure 2: The Fano matroid

For any field $\mathbb{F}$, the class of $\mathbb{F}$-representable matroids is minor-closed, which means that we can hope to characterise the class by explicitly listing the excluded

minors. Such a characterisation has been completed only when $\mathbb{F}$ has size two [57], three [4, 53], or four [24]. In these cases, the numbers of excluded minors are one, four, and seven respectively. The number of excluded minors when $\mathbb{F}$ is the field of five elements is known to be more than 2000 [9], so an explicit excluded-minor characterisation in this next case may be out of reach for some time.

The famous Robertson-Seymour project on graph minors shows that any minor-closed class of graphs has a finite number of excluded minors [50]. This is not necessarily the case for minor-closed classes of matroids: for example, when $\mathbb{F}$ is an infinite field, there are infinitely many excluded minors for the class of $\mathbb{F}$-representable matroids. (Oxley attributes a proof to Geelen and Whittle [47, Theorem 6.5.17].) Rota published the following influential conjecture in 1971 [51].

Conjecture 2.2 (Rota's Conjecture) *Let $\mathbb{F}$ be a finite field. There are only finitely many excluded minors for the class of $\mathbb{F}$-representable matroids.*

A proof of Rota's Conjecture has been announced by Geelen, Gerards, and Whittle [29] as the culmination of a massive project investigating the structure of minor-closed classes of $\mathbb{F}$-representable matroids [25, 27]. These structural results are inspired by the project of Robertson and Seymour that illuminated the structure of minor-closed classes of graphs.

2.3 Biased graphs

Let G be a graph. A *theta subgraph* of G comprises two distinct vertices, u and v, and three paths that join u to v, where u and v are the only vertices that are in more than one of these paths. Thus any theta subgraph contains three cycles. A *linear class* is a set $\mathcal{B}$ of cycles of G such that no theta subgraph contains exactly two cycles in $\mathcal{B}$. In this case, we say $(G, \mathcal{B})$ is a *biased graph*. The cycles in $\mathcal{B}$ are said to be *balanced*. If $(G, \mathcal{B})$ is a biased graph then $F(G, \mathcal{B})$ is the corresponding *frame matroid*. The ground set of this matroid is $E(G)$ and the independent sets are the subsets $X \subseteq E(G)$ such that $G[X]$ contains no balanced cycles and no more than one cycle per connected component. Biased graphs and their related matroids were introduced by Zaslavsky [63, 64]. If every cycle is balanced, the independent sets of the frame matroid are exactly the forests of G, so we have reduced to the case in which $F(G, \mathcal{B})$ is the *graphic matroid* $M(G)$.

Say that $\Omega = (G, \mathcal{B})$ is a biased graph. Let A and B be disjoint subsets of edges such that $G[A]$ is acyclic. Let $\mathcal{P}$ be the set of connected components of $G[A]$. The graph $G/A\backslash B$ has $\mathcal{P}$ as its vertex-set. The edges of $G/A\backslash B$ are the edges of $E(G)\backslash(A \cup B)$ and an edge $e \in E(G)\backslash(A \cup B)$ is incident with a component $P \in \mathcal{P}$ if and only if P contains a vertex v such that v and e are incident in G. Let C be a cycle of $G/A\backslash B$. There is a unique cycle C' of G such that $C'\backslash C \subseteq A$. We declare C to belong to the class $\mathcal{B}/A\backslash B$ if and only if C' is in $\mathcal{B}$. Now $\Omega/A\backslash B = (G/A\backslash B, \mathcal{B}/A\backslash B)$ is a biased graph, and any biased graph that arises in this way is a *minor* of Ω. This definition is constructed in such a way that $F(\Omega/A\backslash B) = F(\Omega)/A\backslash B$ [64, Theorem 2.5]. The definitions of minor-closed classes, excluded minors, antichains, and well-quasi-ordered classes in the context of biased graphs are analogous to the corresponding definitions for matroids.

Let G be a graph and let Γ be a group (written multiplicatively). Let L be the set of tuples $(e, u, v) \in E(G) \times V(G) \times V(G)$ where $\{u, v\}$ is the set of vertices incident with e. A function $\gamma\colon L \to \Gamma$ is a Γ-*gaining* of G if $\gamma(e, u, v) = \gamma(e, v, u)^{-1}$ whenever e is an edge joining distinct vertices u and v. If $W = v_0, e_0, v_1, \ldots, v_{n-1}, e_{n-1}, v_n$ is a walk in G, then we define $\gamma(W)$ to be $\gamma(e_0, v_0, v_1) \cdots \gamma(e_{n-1}, v_{n-1}, v_n)$ and refer to this product as the *gain* of the walk. The gain of a closed walk may depend on the choice of the starting point and the orientation of travel around the walk, but any two gains obtained by different choices will be conjugates in Γ. This means that the class of cycles with gain equal to the identity id_Γ is well defined. We denote this class by $\mathcal{B}_\gamma$. It is easy to see that $\mathcal{B}_\gamma$ is a linear class. If Ω is a biased graph and $\Omega = (G, \mathcal{B}_\gamma)$ for some Γ-gaining γ, then Ω is Γ-*gainable*. A biased graph that is Γ-gainable for at least one group Γ is *gainable*. If γ is a Γ-gaining of G, then $F(G, \mathcal{B}_\gamma)$ is a Γ-*gain-graphic* matroid.

Figure 3 gives an illustration of a gain-graphic matroid. The group Γ in this case is the Klein four-group, isomorphic to the direct product $\mathbb{Z}_2 \otimes \mathbb{Z}_2$. Let id be the identity element and let a, b, and c be the non-identity elements, so that $a+b+c = \mathrm{id}$. On the left side of the diagram, we show a Γ-gained graph. Each edge is labelled with an integer between 1 and 9, as well as an element of Γ. When an edge e is directed from the vertex u to v, and is labelled with $\alpha \in \Gamma$ this indicates that α is the image of (e, u, v) under the gaining. On the right side of the diagram we see a geometric representation of the corresponding Γ-gain-graphic matroid, which has the ground set $\{1, 2, \ldots, 9\}$. Since this matroid has rank three, this diagram represents independent subsets in exactly the same way as Figure 2. We note that $\{1, 2, 3\}$ is the set of loops in the graph, and is a basis of the matroid. Every other element of the matroid is positioned on a line between two of these basis elements, and its location on that line is essentially determined by its gain label. In this way, we can think of the points in a gain-graphic matroid as being given coordinates by elements of a group, in the same way that the points in a representable matroid are given coordinates from a field. So representable and gain-graphic matroids are analogues — both are coordinatised by algebraic structures.

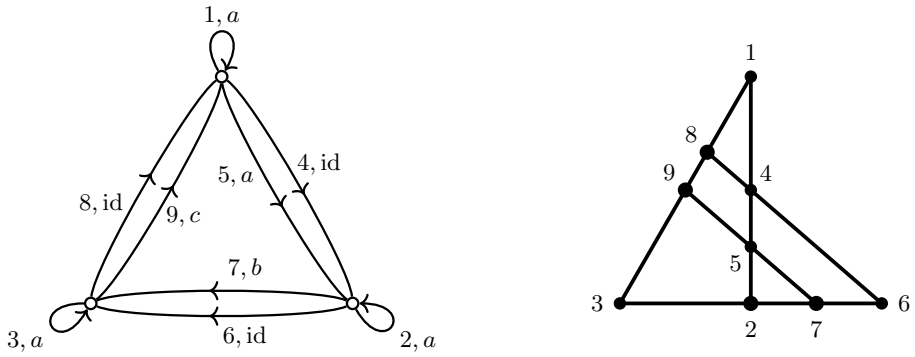

Figure 3: A graph with a gaining and the corresponding gain-graphic matroid

Classes of representable matroids are central in the history of matroid theory. There is a symmetry between classes of representable matroids and classes of gain-graphic matroids. This symmetry can be seen in a theorem by Kahn and Kung [35]

and in the work of Geelen, Gerards, and Whittle [28] that describes the structure of classes of finite-field-representable matroids. These results show that the structure of finite-field-representable classes and finite-group-gain-graphic classes are intrinsically linked. Despite the symmetry of the two types of classes, gain-graphic matroids have not received as much attention as representable matroids. A major theme of this survey will be looking for gain-graphic analogues of theorems that apply to representable matroids.

2.4 Monadic logic

The defining characteristic of monadic second-order logic is that quantifications take place over both elements and subsets of the domain. This distinguishes it from first-order logic (where we may quantify only over elements of the domain) and full second-order logic (where we may quantify over relations of any arity). In this section we will give formal definitions of the syntax and semantics for some variants of monadic second-order logic in the context of discrete structures.

All of the languages we define are recursively constructed from *atomic formulas*. The form that the atomic formulas take depends on the particular language. If φ is a formula, then $\mathrm{Var}(\varphi)$ and $\mathrm{Free}(\varphi)$, respectively, are the sets of *variables* and *free variables* that appear in φ. A free variable is one that has not been quantified over. In addition, we let $\mathrm{Bound}(\varphi) = \mathrm{Var}(\varphi) \setminus \mathrm{Free}(\varphi)$ stand for the set of *bound variables*. Every formula is a sequence of symbols, and the formulas of the language are exactly the sequences that are constructed using a finite application of the following rules.

- Every atomic formula is a formula.
- If φ is a formula, then so is $(\neg\varphi)$. $\mathrm{Var}(\neg\varphi)$ and $\mathrm{Free}(\neg\varphi)$ are respectively equal to $\mathrm{Var}(\varphi)$ and $\mathrm{Free}(\varphi)$.
- If φ_1 and φ_2 are formulas such that $\mathrm{Free}(\varphi_1) \cap \mathrm{Bound}(\varphi_2) = \emptyset = \mathrm{Bound}(\varphi_1) \cap \mathrm{Free}(\varphi_2)$, then $(\varphi_1 \wedge \varphi_2)$ is a formula. $\mathrm{Var}(\varphi_1 \wedge \varphi_2)$ and $\mathrm{Free}(\varphi_1 \wedge \varphi_2)$ are respectively equal to $\mathrm{Var}(\varphi_1) \cup \mathrm{Var}(\varphi_2)$ and $\mathrm{Free}(\varphi_1) \cup \mathrm{Free}(\varphi_2)$.
- If φ is a formula and Z_i is in $\mathrm{Free}(\varphi)$, then $(\exists Z_i \varphi)$ is a formula. $\mathrm{Var}(\exists Z_i \varphi)$ is equal to $\mathrm{Var}(\varphi)$ and $\mathrm{Free}(\exists Z_i \varphi) = \mathrm{Free}(\varphi) \setminus \{Z_i\}$.

We omit parentheses freely from formulas in order to improve readability. We also freely use notations such as $\vee$, $\rightarrow$, $\leftrightarrow$, and $\forall$ with their usual logical meanings, since each of these can be defined in terms $\neg$, $\wedge$, and $\exists$. A formula with no free variables is a *sentence*.

We cannot speak directly about the cardinality of sets in these languages. However, we often allow ourselves atomic formulas that say when the cardinality is congruent to a particular residue modulo some integer q. The set of formulas that are constructed without using any modular cardinality predicates form *monadic second-order logic* (MSO), and the set of all formulas (including those that use modular counting) is *counting monadic second-order logic* (CMSO).

As an example, the logic MSO^{Σ} we deployed in Section 1.1 has atomic formulas of the form $Z_i \subseteq Z_j$, $\mathsf{term}(Z_i)$, and $\mathsf{char}_\alpha(Z_i)$ where α is in the alphabet Σ. If φ is equal to $Z_i \subseteq Z_j$ then $\mathrm{Var}(\varphi) = \mathrm{Free}(\varphi) = \{Z_i, Z_j\}$. If $\varphi = \mathsf{char}_\alpha(Z_i)$ or $\mathsf{term}(Z_i)$, then $\mathrm{Var}(\varphi) = \mathrm{Free}(\varphi) = \{Z_i\}$.

2.4.1 Logic for graphs. The logic $\mathrm{CMSO}_1^{\mathrm{G}}$ has variables $v_1, v_2, v_3, \ldots$ representing vertices, and $V_1, V_2, V_3, \ldots$ representing subsets of vertices. The atomic formulas have the form: (i) $v_i = v_j$, (ii) $v_i \in V_j$, (iii) $\mathsf{adj}(v_i, v_j)$, and (iv) $|V_i| \equiv p \bmod q$, where p and q are integers satisfying $0 \leq p < q$.

In all of these cases, the set of variables in an atomic formula is exactly as one would expect and every variable is free in an atomic formula. Note that $\mathrm{MSO}_1^{\mathrm{G}}$ is the set of formulas constructed without using any formulas of type (iv). The interpretation of $\mathsf{adj}(v_i, v_j)$ is that vertices v_i and v_j are *adjacent* (joined by an edge).

The logic $\mathrm{CMSO}_2^{\mathrm{G}}$ has variables $v_1, v_2, v_3, \ldots$ representing vertices, $e_1, e_2, e_3, \ldots$ representing edges, and variables $V_1, V_2, V_3, \ldots$ and $E_1, E_2, E_3, \ldots$ which respectively represent subsets of vertices and edges. The atomic formulas have the form: (i) $v_i = v_j$ or $e_i = e_j$, (ii) $v_i \in V_j$ or $e_i \in E_j$, (iii) $\mathsf{inc}(v_i, e_j)$, and (iv) $|V_i| \equiv p \bmod q$ or $|E_j| \equiv p \bmod q$, where p and q are integers satisfying $0 \leq p < q$.

We interpret $\mathsf{inc}(v_i, e_j)$ to mean that e_j is incident with v_i. The language $\mathrm{CMSO}_2^{\mathrm{G}}$ is strictly more powerful than $\mathrm{CMSO}_1^{\mathrm{G}}$. For example, the former language can define the class of Hamiltonian graphs and the latter cannot [15, Proposition 5.19]. It is common to denote these languages by CMSO_1 and CMSO_2, or CMS_1 and CMS_2.

2.4.2 Logic for hypergraphs. The logics we have defined up to this point are well established in the literature. Now we move to languages that have not been explored so extensively. Let $\mathcal{P}$ be a finite set of unary predicates. The language $\mathrm{CMSO}(\mathcal{P})$ has $Z_1, Z_2, Z_3, \ldots$ as its variables, and the following atomic formulas: (i) $Z_i \subseteq Z_j$, (ii) $P(Z_i)$, for any predicate $P \in \mathcal{P}$, and (iii) $|Z_i| \equiv p \bmod q$, where p and q are integers satisfying $0 \leq p < q$.

In the list below, we describe some variants of monadic second-order logic for hypergraphs.

(i) $\mathrm{CMSO}_1^{\mathrm{hyp}} = \mathrm{CMSO}(\{\mathsf{hyp}\})$. We will use this as the general monadic second-order logic for hypergraphs.

(ii) $\mathrm{CMSO}_1^{\mathrm{ind}} = \mathrm{CMSO}(\{\mathsf{ind}\})$. We will use this logic in the specific case that the hyperedges are intended to be the independent sets of a matroids. This is the logic introduced by Hliněný [31, 32]. It has been denoted by various notations in the past, including MSOL, CMS_M, and CMS_0. Obviously, we could just as well use logics such as $\mathrm{CMSO}(\{\mathsf{circuit}\})$ or $\mathrm{CMSO}(\{\mathsf{basis}\})$ in the cases where the hyperedges are circuits or bases of matroids. Any of these logics will have equivalent expressive power.

(iii) $\mathrm{CMSO}_1^{\mathrm{star}} = \mathrm{CMSO}(\{\mathsf{star}\})$. In this logic, the variables will represent sets of edges, and the subsets satisfying star will be the *vertex stars* of a graph (the sets of edges that are incident with a vertex).

(iv) $\mathrm{CMSO}_1^{\mathrm{bias}} = \mathrm{CMSO}(\{\mathsf{star}, \mathsf{bal}\})$. This is the logic we use to discuss biased graphs. The sets satisfying bal will be the edge-sets of balanced cycles.

The subscript 1 is a reminder that these logics are 1-*sorted* logics, meaning that there is only one type of variable. This distinguishes the logic from the 2-*sorted* logic $\mathrm{CMSO}_2^{\mathrm{hyp}}$ that we discussed in the introduction.

Now that we have formally defined the syntax of monadic logic for hypergraphs, we will move to the semantics: that is, we define what it means for a formula to be satisfied by a discrete structure. Let $\mathcal{P}$ be a collection of unary predicates. We say that a *$\mathcal{P}$-structure* is a finite set E, along with a collection of subsets $\mathcal{I}_P \subseteq 2^E$ for each predicate $P \in \mathcal{P}$. Let φ be a formula in CMSO($\mathcal{P}$) and let $\theta\colon \text{Free}(\varphi) \to 2^E$ be a function taking free variables to subsets of E. We refer to θ as an *interpretation*. We will define what it means for φ to be *satisfied* by $(E, \{\mathcal{I}_P\}_{P\in\mathcal{P}})$ under the interpretation θ. If φ is the atomic formula $Z_i \subseteq Z_j$, then it is satisfied if $\theta(Z_i)$ is a subset of $\theta(Z_j)$. If φ is $P(Z_i)$, for some P in $\mathcal{P}$, then φ is satisfied if $\theta(Z_i)$ is a member of $\mathcal{I}_P$. And finally, if φ is $|Z_i| \equiv p \bmod q$, then φ is satisfied if the cardinality of $\theta(Z_i)$ is congruent to p modulo q.

Now we define what satisfaction means when φ is not atomic. If φ is $\neg\psi$ for some formula ψ, then φ is satisfied exactly when ψ is *not* satisfied by $(E, \{\mathcal{I}_P\}_{P\in\mathcal{P}})$ under θ. Next assume that φ is a conjunction $\psi_1 \wedge \psi_2$. For $i = 1, 2$, let θ_i be the restriction of θ to Free(ψ_i). Then φ is satisfied exactly when ψ_i is satisfied by $(E, \{\mathcal{I}_P\}_{P\in\mathcal{P}})$ under θ_i, for both $i = 1$ and $i = 2$. Next, if φ is $\exists Z_i\psi$, then φ is satisfied if there is some subset $X \subseteq E$ such that ψ is satisfied by $(E, \{\mathcal{I}_P\}_{P\in\mathcal{P}})$ under the interpretation obtained from θ by adding the ordered pair (Z_i, X). This completes our definition of the semantics of CMSO($\mathcal{P}$).

We illustrate the power of CMSO($\mathcal{P}$) by showing how we can define the structures that most interest us. First we define some useful formulas. The formula $\mathsf{union}(Z_i, Z_j, Z_k)$ is $\forall Z_l\ (\mathsf{sing}(Z_l) \to (Z_l \subseteq Z_k \leftrightarrow (Z_l \subseteq Z_i \vee Z_l \subseteq Z_j)))$, which will be satisfied exactly when Z_k is interpreted as the union of Z_i and Z_j. If P is a predicate in $\mathcal{P}$, then $\mathsf{max}_P(Z_i)$ stands for $P(Z_i) \wedge \forall Z_j\ ((Z_i \subseteq Z_j \wedge P(Z_j)) \to Z_i = Z_j)$. Thus $\mathsf{max}_P(Z_i)$ is satisfied when Z_i is interpreted as a member of $\mathcal{I}_P$ that has no proper superset in $\mathcal{I}_P$.

A matroid (axiomatised via independent sets) is a pair $(E, \mathcal{I}_{\mathsf{ind}})$ satisfying the following $\text{MSO}_1^{\text{ind}}$-sentences.

(**I1**) $\exists Z_1\ \mathsf{ind}(Z_1)$

(**I2**) $\forall Z_1\ \forall Z_2\ ((Z_1 \subseteq Z_2 \wedge \mathsf{ind}(Z_2)) \to \mathsf{ind}(Z_1))$

(**I3′**) $\forall Z_1\ \forall Z_2\ (\mathsf{ind}(Z_1) \wedge \max_{\mathsf{ind}}(Z_2) \wedge \neg \max_{\mathsf{ind}}(Z_1)) \to$
$\exists Z_3\ ((\mathsf{sing}(Z_3) \wedge Z_3 \subseteq Z_2 \wedge Z_3 \nsubseteq Z_1) \wedge \forall Z_4\ (\mathsf{union}(Z_1, Z_3, Z_4) \to \mathsf{ind}(Z_4)))$

Because we are not able to say that one set is larger than another in $\text{MSO}_1^{\text{ind}}$, we have replaced (**I3**) from Section 2.2. The new statement (**I3′**) says that if I and J are independent sets where J is maximal and I is not, then there is an element $e \in J\backslash I$ such that $I \cup \{e\}$ is independent. These axiom schemes are equivalent.

We can think of a graph as being a hypergraph $(E, \mathcal{I}_{\mathsf{star}})$ that satisfies the $\text{MSO}_1^{\text{star}}$-sentences

- $\forall Z_1\ (\mathsf{sing}(Z_1) \to \exists Z_2\ (\mathsf{star}(Z_2) \wedge Z_1 \subseteq Z_2))$
- $\forall Z_1\ \forall Z_2\ \forall Z_3\ \forall Z_4\ (\mathsf{sing}(Z_1) \wedge \mathsf{star}(Z_2) \wedge \mathsf{star}(Z_3) \wedge \mathsf{star}(Z_4) \wedge$
$Z_1 \subseteq Z_2 \wedge Z_1 \subseteq Z_3 \wedge Z_1 \subseteq Z_4) \to (Z_2 = Z_3 \vee Z_2 = Z_4 \vee Z_3 = Z_4)$

With these statements we are saying that every edge needs to be in either one or two vertex stars. Now we can think of the sets satisfying star as being vertices, and the

elements of the ground set as being edges that joins those vertices, where incidence of an edge and a vertex is just element inclusion. It is interesting that this is the logical language for graphs that arises naturally in some matroid contexts, given that it de-emphasises vertices and emphasises edges. This recalls Tutte's comment: "If a theorem about graphs can be expressed in terms of edges and circuits only it probably exemplifies a more general theorem about matroids" [58, pp. 497].

We can also use $\mathrm{MSO}_1^{\mathrm{bias}}$ to define biased graphs. From [21, Proposition 3.6] we have a $\mathrm{MSO}_1^{\mathrm{bias}}$-formula cycle with one free variable, which is satisfied when that variable is sent to the set of edges in a cycle. From this we construct a formula theta with a free variable, which is satisfied when the variable is sent to the edges of a theta subgraph: a set X of edges is a theta subgraph precisely when it contains exactly three distinct cycles and any proper subset of X contains at most one cycle. To define a bias graph, we start with the $\mathrm{MSO}_1^{\mathrm{star}}$-formulas above, and we add a sentence saying that if Z_1, Z_2, and Z_3 are subsets satisfying cycle, where their union satisfies theta, then it is not the case that exactly two of Z_1, Z_2, and Z_3 satisfy bal.

Hliněný observed that for any matroid N, there is an $\mathrm{MSO}_1^{\mathrm{ind}}$-sentence that is satisfied exactly by the matroids that have a minor isomorphic to N [32, Lemma 5.1]. We sketch a proof of an analogous fact for biased graphs, using [21, Proposition 3.5], which gives us a formula satisfied by connected subsets of edges.

Proposition 2.3 *Let Ω be a biased graph. There is an $\mathrm{MSO}_1^{\mathrm{bias}}$-sentence that is satisfied exactly by the biased graphs that have a minor isomorphic to Ω.*

Sketch proof. We assume $\Omega = (G, \mathcal{B})$ has vertices $v_1, \ldots, v_n$ and edges $e_1, \ldots, e_m$. We build a sentence that says there exist connected subsets $F_1, \ldots, F_n$ of edges such that none of these sets contains a cycle, and no vertex star contains edges from more than one of these sets. This establishes the n trees that will correspond to the vertices of a Ω-minor. Next we assert that there exist singleton sets $E_1, \ldots, E_m$ of edges such that the edge in E_i is incident with a vertex in $G[F_j]$ if and only if e_i is incident with v_j. The final step involves a separate clause for each cycle C of Ω. There will exist a unique cycle that contains the edge-set $\cup_{e_i \in C} E_i$, and this cycle must be balanced exactly when C is balanced in Ω. □

From Hliněný's observation, we see that any minor-closed class of matroids with finitely many excluded minors can be defined by an $\mathrm{MSO}_1^{\mathrm{ind}}$-sentence. The class of *lattice-path matroids* provides an example of a natural minor-closed class that is $\mathrm{MSO}_1^{\mathrm{ind}}$-definable [34, Theorem 5.7], despite the fact that it has infinitely many excluded minors [8].

2.5 Tree automata

The BET Theorem implies there is a fast algorithm for testing membership in any language defined by a monadic sentence. This is because running a finite-state automaton on a string takes linear time. Courcelle's Theorem and its descendants rely on the same idea. However graphs and hypergraphs do not typically have a linear ordering in the same way that a string does. So it is more natural to use automata that process trees. In this section we explain how hypergraphs can be described by tree-shaped data structures.

We describe a rooted binary tree where we distinguish between left and right children. Let T be a *tree*, meaning a connected graph without cycles. We distinguish a *root vertex*, denoted by root. Any vertex with degree one is a *leaf*. The set of leaves of T is denoted by $L(T)$. We let $I(T)$ be the set of non-leaf vertices, which we say are *internal*. We require that every non-root internal vertex has degree three and that the root has degree two. Every vertex $v \in I(T)$ has two neighbours that are not contained in the path from v to root. We say that these two neighbours are the *children* of v. We distinguish the *left* and *right* children of v using a bijection c_v from the set of children of v to the set $\{L, R\}$.

Let Q be a finite set of *states* and let $\mathsf{acc} \subseteq Q$ be the set of *accepting states*. Let s be a non-negative integer, which we will call the *arity*. To each internal vertex v, we associate a *transition relation* τ_v. If both the children of v are leaves, then τ_v is a subset of $\{\checkmark, \text{✗}\}^s \times \{\checkmark, \text{✗}\}^s \times Q$. If neither child is a leaf, then τ_v is a subset of Q^3. If the left child of v is a leaf and the right child is not, then τ_v is contained in $\{\checkmark, \text{✗}\}^s \times Q^2$, and in the final case, where the right child is a leaf but the left child is not, then τ_v is a subset of $Q \times \{\checkmark, \text{✗}\}^s \times Q$. (We use the convention that $\{\checkmark, \text{✗}\}^0 = \{\epsilon\}$.) We refer to the entire system

$$\mathbf{T} = (T, \mathsf{root}, \{c_v\}_{v \in I(T)}, s, Q, \mathsf{acc}, \{\tau_v\}_{v \in I(T)})$$

as a *tree automaton*.

Let S be a function from $L(T)$ to $\{\checkmark, \text{✗}\}^s$. A *run* of $\mathbf{T}$ on S is a function $r\colon I(T) \to Q$ such that for any vertex $v \in I(T)$ with left child v_L and right child v_R, the following conditions hold: (i) if v_L and v_R are leaves, then $(S(v_L), S(v_R), r(v))$ is in τ_v, (ii) if neither v_L nor v_R is a leaf, then $(r(v_L), r(v_R), r(v))$ is in τ_v, (iii) if v_L is a leaf but v_R is not, then $(S(v_L), r(v_R), r(v))$ is in τ_v, and (iv) if v_R is a leaf but v_L is not, then $(r(v_L), S(v_R), r(v))$ is in τ_v. A run r is *accepting* if $r(\mathsf{root})$ is in acc. If there exists an accepting run on S, then we say that $\mathbf{T}$ *accepts* S. We define $\mathbf{T}$ to be *deterministic* if for every choice of the function S, there is a unique run of $\mathbf{T}$ on S.

Any function $S\colon L(T) \to \{\checkmark, \text{✗}\}$ picks out a subset of $L(T)$, which we denote by $L(S)$. To be precise, $l \in L(T)$ belongs to $L(S)$ if and only if $S(l) = \checkmark$. More generally, if s is a positive integer and S is a function from $L(T)$ to $\{\checkmark, \text{✗}\}^s$, then for any integer $j \in \{1, \ldots, s\}$, we set $L_j(S) \subseteq L(T)$ to be the set such that l is in $L_j(S)$ if and only if the coordinate in position j of $S(l)$ is $\checkmark$. We identify $\{\checkmark, \text{✗}\}$ with $\{\checkmark, \text{✗}\}^1$, so $L(S)$ is the same as $L_1(S)$.

Let $\mathbf{T}$ be a tree automaton with arity 1. Then we can think of $\mathbf{T}$ as accepting some of the subsets of $L(T)$. We let $\mathcal{I}(\mathbf{T})$ be

$$\{L(S)\colon S \text{ is a function from } L(T) \text{ to } \{\checkmark, \text{✗}\},\ \mathbf{T} \text{ accepts } S\}.$$

A tree automaton $\mathbf{T} = (T, \mathsf{root}, \{c_v\}_{v \in I(T)}, 1, Q, \mathsf{acc}, \{\tau_v\}_{v \in I(T)})$ with arity 1 corresponds to the hypergraph $(L(T), \mathcal{I}(\mathbf{T}))$. This will be our primary method for describing a hypergraph for the purposes of computation. It is a fairly easy exercise to see that any hypergraph can be described as $(L(T), \mathcal{I}(\mathbf{T}))$ for some choice of automaton $\mathbf{T}$. The question is, how many states are required for a tree automaton that will describe the hypergraph? This becomes a measure of its structural complexity. In the next section, we will discuss the fact that a class of hypergraphs can

be described by tree automata with a bounded number of states if and only if the class has bounded *decomposition-width*.

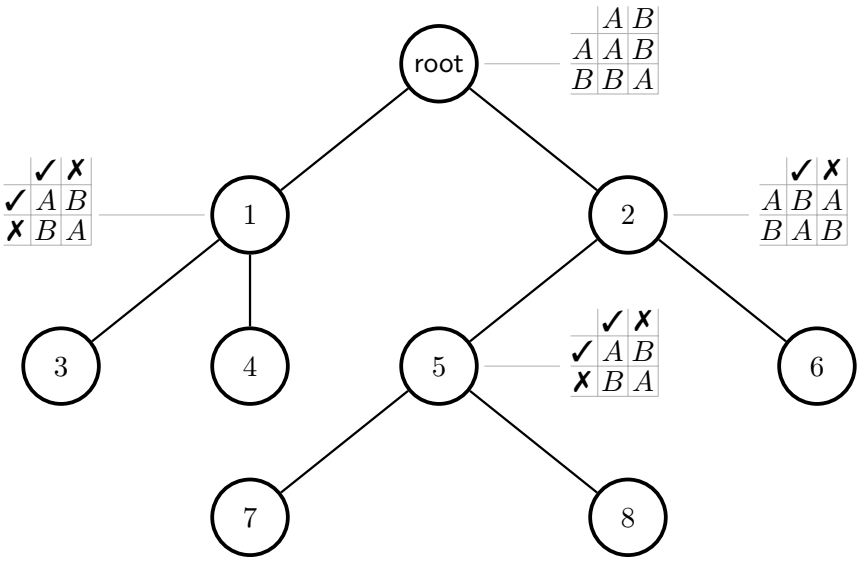

Figure 4: A tree automaton

Example 2.4 Figure 4 illustrates a tree automaton $\mathbf{T}$. In this case the arity s is 1, and Q is the set $\{A, B\}$. Let us define acc to be $\{A\}$. Each internal vertex v is labelled with a 2×2 table. The transition relation τ_v is exactly the set of triples (α, β, γ) where γ appears in the table in the row labelled by α and the column labelled by β. Note that $\mathbf{T}$ is deterministic. Let $S\colon L(T) \to \{\checkmark, \boldsymbol{\times}\}$ be the function such that $L(S) = \{3, 6, 7\}$. Then $\mathbf{T}$ has a unique run r on S and r acts as follows: $r\colon 5 \mapsto B \quad r\colon 2 \mapsto A \quad r\colon 1 \mapsto B \quad r\colon \mathsf{root} \mapsto B$. So $\mathbf{T}$ does not accept S. In fact, we can easily see that $\mathcal{I}(\mathbf{T})$ is exactly the collection of subsets of $L(T)$ that have even cardinality.

The next result is proved using the same boot-strapping approach as Lemma 1.7 (see [22, Lemma 4.7]). In the language of Section 6, it shows that definability implies recognisability for hypergraphs of bounded decomposition-width.

Lemma 2.5 *Let φ be any formula in $\mathrm{CMSO}_1^{\mathrm{hyp}}$ and let $Z_{i_1}, Z_{i_2}, \ldots, Z_{i_s}$ be the free variables of φ. Let $\mathbf{T} = (T, \mathsf{root}, \{c_v\}_{v \in I(T)}, 1, Q, \mathsf{acc}, \{\tau_v\}_{v \in I(T)})$ be a tree automaton and let M be the hypergraph $(L(T), \mathcal{I}(\mathbf{T}))$. There exists a tree automaton $\mathbf{T}_\varphi = (T, \mathsf{root}, \{c_v\}_{v \in I(T)}, s, Q_\varphi, \mathsf{acc}_\varphi, \{\rho_v\}_{v \in I(T)})$ such that for any function $S\colon L(T) \to \{\checkmark, \boldsymbol{\times}\}^s$, we have that $\mathbf{T}_\varphi$ accepts S if and only if φ is satisfied by M under the interpretation that takes each Z_{i_j} to $L_j(S)$.*

2.6 Decomposition-width and branch-width

Numerical *width parameters* are fundamental in modern structural discrete mathematics. Algorithmic meta-theorems such as Courcelle's Theorem often require that the input class has bounded complexity, meaning that some width parameter is

bounded for objects in the class. In the graph context, *tree-width* and *clique-width* are commonly used. For matroids, the most frequently used width parameter is *branch-width*, which we now define. Let T be a tree such that every vertex has degree one or three and let $L(T)$ be the set of leaves of T. Let $M = (E, \mathcal{I})$ be any hypergraph and let π be a bijection from E to $L(T)$ — we say that (T, π) is a *tree decomposition* of M. Assume that e is an edge joining vertices u and v in T. Then e partitions E into sets U_e and V_e, where U_e is the set of elements $x \in E$ such that the path from $\pi(x)$ to u does not contain v. Assuming that M is a matroid, for any subset $X \subseteq E$, the *width* of e is the numerical parameter $r_M(U_e)+r_M(V_e)-r(M)+1$. The branch-width of M is the smallest value k such that M has a tree decomposition where every edge has width at most k. The branch-width of a matroid M is bounded above by $r(M) + 1$.

Decomposition-width uses the same framework, but it applies to all hypergraphs and does not require a matroidal rank function. We change the definition of the width of an edge. Say that J is any subset of E. We define an equivalence relation on subsets of J. Say that $X, X' \subseteq J$ are equivalent if for any subset $Y \subseteq E \backslash J$, either both $X \cup Y$ and $X' \cup Y$ are in $\mathcal{I}$, or neither one is. The width of the edge e in T is the maximum number of equivalence classes when J is equal to either U_e or V_e. Now the decomposition-width of M is the smallest number k such that there is a tree decomposition of M where every edge has width at most k. The terminology of decomposition-width was first used by Král [38] and Strozecki [55]. The definition I have given here differs from theirs, but is equivalent in the sense that when $\mathcal{M}$ is a class of hypergraphs, the maximum decomposition-width of hypergraphs in $\mathcal{M}$ is finite under both definitions, or under neither. The same comment applies to the parameter *hyper-rankwidth*, introduced by Bojańczyk [6].

If a class of matroids has bounded decomposition-width, then it has bounded branch-width, but the converse does not hold [22, Corollary 2.8]. Theorem 1.4 in [22] shows that a class $\mathcal{M}$ of hypergraphs has bounded decomposition-width if and only if there is an integer k such that every $M \in \mathcal{M}$ can be described as $(L(T), \mathcal{I}(\mathbf{T}))$ for some arity-1 tree automaton $\mathbf{T}$ with at most k states. So this becomes our notion of a class of hypergraphs with bounded structural complexity — a class that can be described via tree automata with a bounded number of states.

For the purposes of applying tree automata to test membership of monadically-defined classes, we would like to be able to produce a tree-automata description of a matroid, given some description via a matrix or biased graph. Assuming that the input is restricted to a class with bounded branch-width, Theorem 6.5 in [22] provides us with an algorithm for doing so. The foundation of this algorithm is work by Oum and Seymour [46, Corollary 7.2] that produces tree decompositions for matroids, relative to branch-width. Their algorithm either produces a tree decomposition where every edge has width at most $3\lambda + 1$, or it gives a certificate that the matroid has branch-width greater than λ.

2.7 Transductions

Transductions are one of the most useful tools in the monadic theory of discrete structures. Multiple definitions of transductions are found in the literature [6, 7, 15]. All of them are technical, but express the same intuition. Rather than diving into

another formal definition, I will attempt to explain this intuition with examples. The formal definition I have in mind is tailored to hypergraphs, and can be found in [34].

We think of a transduction as a relation which takes as input a class of structures relative to some language, and produces as output some other class (relative to a possibly different language). Such a relation allows us to use the language of the input class to make statements about objects in the output class. In our conception, we will have a formula dom (for domain), and every satisfying assignment of this formula will correspond to an output of the transduction. The elements of the ground set in the output hypergraph will be the subsets that satisfy a further formula, univ.

We give an example from matroid theory that takes each matroid to its dual. The input and output languages will be $\mathrm{CMSO}_1^{\mathrm{ind}}$. Let dom be a sentence which is satisfied by $M = (E, \mathcal{I})$ when $\mathcal{I}$ is the collection of independent sets of a matroid. Let univ be the formula sing, which is satisfied by singleton sets. Let coindep be a formula with one free variable that is satisfied when that variable is interpreted as a coindependent set. The trio dom, univ, coindep comprises the transduction. In this case, there will be exactly one output for every input that is a matroid. The ground set of the output will be $\{\{e\} \colon e \in E\}$, which we can identify with E. The hyperedges of the output will be the coindependent sets of M, which is to say that the output will be isomorphic to the dual matroid M^*.

For another example, we consider the transduction taking each matroid to its connected components. Let $M = (E, \mathcal{I})$ be a matroid. A *separator* of M is a set X such that $r_M(X) + r_M(E \backslash X) = r(M)$. A non-empty separator that does not properly contain a non-empty separator is a *connected component*. In this example we let dom be a formula with a free variable Z_i such that dom is satisfied when $M = (E, \mathcal{I})$ is a matroid and Z_i is interpreted as a connected component. The formula univ has a free variable, and will be satisfied when this variable is interpreted as a singleton subset of Z_i. So the ground set of the output will be $\{\{e\} \colon e \in Z_i\}$, which we can identify with Z_i. Let indep be the formula that is satisfied when its free variable is interpreted as an independent set. The transduction comprising dom, univ, indep takes each matroid to its connected components.

We can use a transduction to simplify the problem of finding a sentence that defines a particular property by using the *Backwards Translation Theorem*. Say that a hypergraph property is defined by a monadic sentence φ. The Backwards Translation Theorem says that for any transduction, there exists a sentence that will be satisfied by the hypergraph M exactly when the image of M under the transduction contains at least one hypergraph satisfying φ. (Alternatively, when every hypergraph in the image satisfies φ.) So for example, say that we wish to show that a matroid property is monadically-definable. Assume this property holds for a matroid whenever all of its connected components have the property. We have noted that there is a transduction taking each matroid to its connected components, so the Backwards Translation Theorem means that we need only show that the property is monadically-definable for connected matroids. The theorem has become an essential tool in the investigation of recognisability and definability, as we will discuss in Section 6. Versions of the backwards translation theorem can be found in [5, Proposition 20], [7, Lemma B.1], [15, Theorem 1.40], and [34].

3 Complexity

Tree automata provide the engine that powers Courcelle's Theorem and its descendants. These theorems give us *fixed-parameter tractable* algorithms. An algorithm of this type receives input with an associated numerical parameter, λ, and runs in time $O(f(\lambda)n^c)$, where $f(\lambda)$ is a value that depends only on λ. Here n is the size of the input and c is a positive constant. A fixed-parameter tractable algorithm is more desirable than one with running time $O(n^{f(\lambda)})$, because it will typically be useful for a wider range of parameter values. See [16] for an introduction to parameterized complexity.

Theorem 3.1 (Courcelle's Theorem) *Let φ be a $\mathrm{CMSO}_2^{\mathrm{G}}$-sentence. There is a fixed-parameter tractable algorithm for testing φ in graphs with respect to the parameter of tree-width.*

Hliněný introduced the matroid logic $\mathrm{CMSO}_1^{\mathrm{ind}}$ when proving the following result, an analogue of Courcelle's Theorem for finite-field-representable matroids [31, Corollary 5.2].

Theorem 3.2 *Let φ be a $\mathrm{CMSO}_1^{\mathrm{ind}}$-sentence. There is a fixed-parameter tractable algorithm for testing φ with respect to the parameter of branch-width, when the input consists of $\mathbb{F}$-representable matroids (and $\mathbb{F}$ is a finite field).*

Král [38], Strozecki [55], and Funk, Mayhew, and Newman [22] all present abstract versions of Hliněný's Theorem, meaning that they extend the result to matroid classes that are not necessarily representable. All of these theorems rely on statements similar to Lemma 2.5 to show that monadically definable properties can be recognised by tree automata. Results in [23] shows that Hliněný's Theorem applies to other classes that arise naturally in matroid theory: in particular, the class of Γ-gain-graphic matroids, when Γ is a finite group.

We can certify that the hypotheses of Theorem 3.2 are tight by relaxing them and showing that testing some monadic property becomes NP-hard. This is exactly what Hliněný accomplishes in [30, Theorem 5.6], where he shows that it is NP-complete to test whether a matroid contains a particular fixed minor when the input consists of matroids represented over the rational numbers (even when the input is restricted to matroids with branch-width three). Thus we cannot relax the finiteness of $\mathbb{F}$ in Theorem 3.2. A technique from [41] essentially develops a transduction from the graph language $\mathrm{CMSO}_2^{\mathrm{G}}$ to the language $\mathrm{CMSO}_1^{\mathrm{ind}}$ applied to rank-three matroids. Because every such matroid has branch-width at most four, this transduction allows us to show that we cannot relax the condition of bounded branch-width in Theorem 3.2. It should be easy enough to show that similar results apply in the case of Γ-gain-graphic matroids (Conjecture 7.3).

4 Decidability

In this section we talk about classes with *decidable theories*. We discussed this idea in Section 1.1. Say that $\mathcal{A}$ is a logical language which can make statements about a class of structure. This class has a decidable $\mathcal{A}$-theory if there is a theorem-testing machine: that is, a Turing machine which will take as input any sentence

in the language and halt in finite time while deciding whether or not the sentence is satisfied by every structure in the class. The idea of a theorem-testing machine belongs squarely to Hilbert's philosophical program of the first part of the twentieth century. This program was devastated by the discoveries of Gödel, Tarski, Turing, Post, and Church in the 1930s. Because of those discoveries, we now know that classes with decidable theories should be regarded as exceptional: decidability is strong evidence that a class is tractable and well-behaved. The next theorem beautifully characterises the boundary between monadic decidability and undecidability for minor-closed classes of graphs [52].

Theorem 4.1 (Seese's Theorem) *Let $\mathcal{G}$ be a minor-closed class of graphs. The $\mathrm{CMSO}_2^{\mathrm{G}}$-theory of $\mathcal{G}$ is decidable if and only if $\mathcal{G}$ has bounded tree-width.*

Seese's Theorem is an extraordinary piece of work. I find the manner in which it is proved even more amazing than the statement of the theorem. The proof depends upon the structural results of Robertson and Seymour, in particular the Grid Theorem [49]. This theorem tells us that a minor-closed class of graphs is either tree-like (in the sense that it has bounded tree-width), or contains arbitrarily large square grids as minors. In the first case, the minor-closed family is recognised by a tree automaton, and we can prove the "if" direction by using an analogue of the pumping lemma for tree automata. Hliněný and Seese [33] attribute this direction to Courcelle, and separately to Arnborg, Lagergren, Seese. In the second case the class contains arbitrarily large grids, and these are exactly the structures we need to encode the evolution of a computation being carried out by a Turing machine. (Seese does this indirectly by instead encoding a tiling problem.) We think of each row of the grid as encoding the state of the machine tape at one step of its computation. This encoding of computation means that the monadic-theory of the class cannot possibly be decidable, or else we would be able to solve the Halting Problem in finite time. So the structures provided to us by Robertson and Seymour's theory are exactly the structures that correspond to the two strategies we need to prove decidability or undecidability. I cannot see any trivial reason for this correspondence to be so precise — it is one of the most striking mathematical facts I know.

It would be lovely to find an analogue of Seese's Theorem for general minor-closed classes of matroids (Question 7.6). I believe this will be extremely hard, maybe impossible. However, Hliněný and Seese [33, Theorem 6.2] have proved a version for $\mathbb{F}$-representable matroids when $\mathbb{F}$ is finite. This theorem is stated in terms of branch-width rather than tree-width. For the "only if" direction of their theorem, they rely on a matroidal version of the Grid Theorem due to Geelen, Gerards, Whittle [26].

The finiteness of the field $\mathbb{F}$ is necessary for the theorem by Hliněný and Seese. Assume that $\mathbb{F}$ is infinite. The antichain of rank-3 matroids presented in [47, Example 14.1.2] is contained in the class of $\mathbb{F}$-representable matroids. If we take any subset of this antichain and close it under minors, we obtain a class with bounded branch-width. There will therefore be uncountably many such minor-closed classes. Since there are only countably many Turing machines, it follows easily that uncountably many of these classes have undecidable theories.

We note that the analogue of Seese's Theorem for general minor-closed classes of matroids does not hold when we replace tree-width with either branch-width or

decomposition-width. We just saw how to construct uncountably many minor-closed classes that have undecidable monadic theories, despite having bounded branch-width. These classes also have bounded decomposition-width. So neither bounded branch-width nor bounded decomposition-width is enough to guarantee a decidable monadic theory in general minor-closed classes of matroids. Nor is it true that a class with a decidable monadic theory must have bounded branch-width or decomposition-width. The counterexample here is the class of *uniform* matroids. These are the matroids where every circuit is spanning. It is easy to see from [22, Corollary 2.8] that the class of uniform matroids has unbounded decomposition-width, and it also has unbounded branch-width. Despite this, the class of uniform matroids has a decidable $\mathrm{CMSO}_1^{\mathrm{ind}}$-theory (personal communication from Nathan Bowler). So currently I do not know of a statement that would plausibly serve as an analogue of Seese's Theorem for general matroid classes and I am somewhat sceptical that such an analogue exists.

The decidability of the $\mathrm{CMSO}_1^{\mathrm{ind}}$-theory for uniform matroids illustrates an important way in which monadic logic for hypergraphs differs from monadic logic with fixed-arity predicates. Seese has conjectured that in the latter type of logic, when a class of structures has a decidable monadic theory, there is a transduction from this class to the class of trees ([52], see also [33, Conjecture 4.12]). This conjecture remains open. There is nothing tree-like about the class of uniform matroids, despite the fact that the $\mathrm{CMSO}_1^{\mathrm{ind}}$-theory of uniform matroids is decidable. So there is no obvious analogue of Seese's Conjecture for matroids, and this is one reason that it will be difficult to characterise the minor-closed classes of matroids with decidable monadic theories.

Now we will look at analogues for gain-graphic matroids of the work we have just discussed. Let Γ be a finite group, and let $\mathcal{M}$ be a minor-closed class of Γ-gain-graphic matroids. We would like to know that the $\mathrm{CMSO}_1^{\mathrm{ind}}$-theory of $\mathcal{M}$ is decidable if and only if $\mathcal{M}$ has bounded branch-width. I certainly believe this to be the case (Conjecture 7.4). However it is not possible to assemble either direction of the statement from results currently in the literature. Proving the "if" direction could be accomplished by applying a tree-automaton version of the pumping lemma to classes of Γ-gain-graphic matroids with bounded branch-width. I am sure that a minor-closed class of frame matroids with unbounded branch-width contains arbitrarily large grids among its underlying graphs (but I not sure that this proof has been written down). I believe that any class with arbitrarily large grids will have an undecidable monadic theory (Conjecture 7.5). This feels quite provable, and should be enough to establish the "only if" direction of the gain-graphic analogue of the theorem by Hliněný and Seese.

We close this section by noting a result from [22, Theorem 7.4]. The proof relies on the tree automaton version of the pumping lemma.

Lemma 4.2 *Let $\mathcal{M}$ be a $\mathrm{CMSO}_1^{\mathrm{hyp}}$-definable class of hypergraphs. If $\mathcal{M}$ has bounded decomposition-width, then the $\mathrm{CMSO}_1^{\mathrm{hyp}}$-theory of $\mathcal{M}$ is decidable.*

5 Definability

The results in Sections 3 and 4 tell us that if a class of hypergraphs is definable in $\mathrm{CMSO}_1^{\mathrm{hyp}}$, we gain access to powerful tools from the theory of computation. So we are well motivated to ask which classes are definable in this way.

We have already discussed Hliněný's observation that a minor-closed class of matroids is $\mathrm{MSO}_1^{\mathrm{ind}}$-definable if it has a finite number of excluded minors. The resolution of Rota's Conjecture (Conjecture 2.2) therefore means that the class of $\mathbb{F}$-representable matroids is $\mathrm{MSO}_1^{\mathrm{ind}}$-definable when $\mathbb{F}$ is a finite field. The other direction of the following theorem is due to Mayhew, Newman, and Whittle [42].

Theorem 5.1 *Let $\mathbb{F}$ be a field. The class of $\mathbb{F}$-representable matroids is MSO_1-definable if and only if $\mathbb{F}$ is finite.*

We will explain how we prove the "only if" direction of this theorem: that is, how we prove non-definability of a class of hypergraphs. The idea is similar to the characterisation of regular languages developed by Myhill [43] and Nerode [45]. Their characterisation is built on an equivalence relation that uses the concatenation operation on strings. For our analogue, we need a method to "concatenate" hypergraphs. The techniques in [42] apply to particular matroidal methods of concatenation. These techniques are generalised by Funk, Matthews, and Mayhew [20] to methods for "gluing" together two structures in such a way that the output is a hypergraph.

Definition 5.2 Let U, V, and C be finite sets such that $U \cap V = \emptyset$. Let c be a function from 2^U to C, and let d be a function from $2^V \times C$ to $\{\checkmark, \boldsymbol{\times}\}$. We define $(U, c) \boxplus (V, d)$ to be the hypergraph

$$(U \cup V, \{X \cup Y : X \subseteq U,\ Y \subseteq V,\ d(Y, c(X)) = \checkmark\}).$$

Definition 5.3 Let $\mathcal{M}$ be a class of hypergraphs and let C be a finite set. When U_1 and U_2 are finite sets and c_i is a function from 2^{U_i} to C for each i, we write $(U_1, c_1) \sim_{\mathcal{M},C} (U_2, c_2)$ to mean that

$$(U_1, c_1) \boxplus (V, d) \in \mathcal{M} \leftrightarrow (U_2, c_2) \boxplus (V, d) \in \mathcal{M}$$

for every finite set V such that $(U_1 \cup U_2) \cap V = \emptyset$ and every function d from $2^V \times C$ to $\{\checkmark, \boldsymbol{\times}\}$.

It is easy to see that the relation $\sim_{\mathcal{M},C}$ is an equivalence relation. The next result [20] provides a hypergraph analogue of the Myhill-Nerode lemma.

Lemma 5.4 *Let $\mathcal{M}$ be a $\mathrm{CMSO}_1^{\mathrm{hyp}}$-definable class of hypergraphs. Let C be a finite set. The equivalence relation $\sim_{\mathcal{M},C}$ has only finitely many equivalence classes.*

We show how to apply the lemma in the next result. An *exact cover* of the hypergraph $(E, \mathcal{I})$ is a subfamily $\mathcal{I}' \subseteq \mathcal{I}$ such that every element of E belongs to exactly one member of $\mathcal{I}'$.

Proposition 5.5 *There is no sentence in $\mathrm{CMSO}_1^{\mathrm{hyp}}$ that defines the class of hypergraphs with exact covers.*

Proof Let $\mathcal{M}$ be the class of hypergraphs with exact covers. For each positive integer s, let U_s and V_s be finite sets with cardinality s. We assume $U_s \cap V_t = \emptyset$ for each pair s and t. Let C be the set $\{1, 2\}$, and for each s, let $c_s \colon 2^{U_s} \to C$ be the function that takes every singleton subset of U_s to 1 and every other subset to 2. Let $d_s \colon 2^{V_s} \times C \to \{\checkmark, \text{✗}\}$ be the function that takes the pair (Y, i) to ✓ when $Y \subseteq V_s$ is a singleton set and $i = 1$. Every other pair in $2^{V_s} \times C$ is taken to ✗. We can see that $(U_s, c_s) \boxplus (V_t, c_t)$ is the complete bipartite graph $K_{s,t}$ where the two sides of the bipartition are U_s and V_t. Now this hypergraph has an exact cover if and only if $s = t$ (because $K_{s,t}$ has a perfect matching if and only if $s = t$). So $\sim_{\mathcal{M},C}$ has one equivalence class for each positive integer s. Lemma 5.4 implies the result. □

We mention some other techniques which have been used to prove non-definability results. Ehrenfeucht–Fraïssé (EF) games are often used to show that a property cannot be defined in first-order logic. We give a very informal description of these games. They feature two antagonists, Splitter and Duplicator. Splitter attempts to distinguish between two structures by making a sequence of choices. Each choice interprets a variable as an element of one of the structures. Duplicator attempts to show that for any choice made by Splitter, it is possible to interpret that variable in the other structure, in such a way that at the end of the game the two interpretations appear identical. If we can prove that Duplicator has a winning strategy, then we have proved a non-definability result.

It appears to be extremely difficult to use EF games to show that a property is not definable in monadic second-order logic, because in the monadic context, Splitter is able to interpret variables not only as elements of the structure, but also as subsets. This provides Splitter with a huge amount of additional power, and makes it much more difficult to prove that Duplicator can always defeat Splitter. Libkin gives a proof that the class of hypergraphs with even-cardinality ground sets is not $\mathrm{CMSO}_1^{\mathrm{hyp}}$-definable [39, Proposition 7.12], and it seems unlikely that we will be able to use EF games to prove non-definability results much more complicated than this.

Van Bevern et al. [1, Corollary 3.16] develop a Myhill-Nerode method for hypergraphs which is somewhat similar to Lemma 5.4. The important difference is that their "gluing" operations produce hypergraphs where only a bounded number of hyperedges can contain elements from both terms in the operation. This contrasts with the operation in Definition 5.2, which allows arbitrarily many such hyperedges. (For example, in the proof of Proposition 5.5, the gluing operation produces $K_{s,t}$, which has st edges crossing between the sets U_s and V_t.) Many natural matroid operations will have an unbounded number of independent sets that cross the join, which makes Lemma 5.4 more useful in the matroid context.

Kotek and Makowsky use the method of *connection matrices* to prove non-monadic-definability of graph parameters [37]. Using their language, the operation $\boxtimes$ that we use in Proposition 5.5 is not *smooth*, so we are not able to use their techniques for that result, although the use of connection matrices to prove non-definability for some variants of 2-sorted hypergraph logic remains an intriguing possibility.

Theorem 5.1 provides us with a clean dichotomy for the monadic definability of the class of $\mathbb{F}$-representable matroids: if $\mathbb{F}$ is finite the class is definable, and other-

wise it is not. It is natural to ask the parallel question for classes of gain-graphic matroids. For what groups Γ is the class of Γ-gain-graphic matroids $\mathrm{MSO}_1^{\mathrm{ind}}$-definable? Daryl Funk and I are currently engaged in a lengthy project to investigate the different ways in which biased graphs can represent frame matroids. Using the language of this survey, this work will show that there is a $\mathrm{CMSO}_1^{\mathrm{ind}}$-sentence φ and a transduction from $\mathrm{CMSO}_1^{\mathrm{ind}}$ to $\mathrm{CMSO}_1^{\mathrm{bias}}$ which will take any frame matroid *not* satisfying φ to the biased graphs that represent it. The sentence φ characterises certain pathological families. Work in preparation by myself, Ben Shahar, and Funk shows that if Γ is a finite group, then the Γ-gainable biased graphs are $\mathrm{CMSO}_1^{\mathrm{bias}}$-definable. Combining these pieces of work and applying the Backwards Translation Theorem shows that the class of Γ-gain-graphic matroids is $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable when Γ is finite. So one direction of the dichotomy continues to hold in the context of gain-graphic matroids.

The other direction is false! There are infinite groups Γ such that the class of Γ-gain-graphic matroids is $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable. The same group of authors will show that if F is a finite group and $\Gamma = \otimes_{i\geq 1} F$ is an infinite direct product, then the class of Γ-gain-graphic matroids is $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable. So Conjecture 1.5 in [21] is false.

A group Γ is *uniformly locally-finite* if there is a function g_Γ such that any subgroup of Γ generated by at most k elements has size at most $g_\Gamma(k)$. Funk, Matthews, and Mayhew use Lemma 5.4 to prove that if Γ is an infinite group and Γ is *not* uniformly locally-finite, then the class of Γ-gain-graphic matroids is *not* $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable. Despite this progress in both the positive and negative directions, it is currently not clear which infinite groups Γ have the property that the class of Γ-gain-graphic matroids is $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable. Nonetheless, I remain hopeful that we will ultimately be able to resolve this problem (Question 7.7).

6 Recognisability vs. definability

When we say a class of structures is *recognisable*, we mean there is some computing machine that accepts structures in the class and rejects structures not in the class. A class is *definable* if it can be characterised by a sentence in some formal logical language. The connection between recognisability and definability is mysterious and has attracted much attention. The canonical result that relates these properties is the BET Theorem, which says that a language of strings is recognised by a finite-state automaton if and only if it can be defined by an MSO^{Σ}-sentence.

Proving that definability implies recognisability is often fairly easy: we must show that for each sentence, there is some machine that recognises the structures satisfying that sentence. This is what we do in Lemmas 1.7 and 2.5. The other direction is often hard. The issue with proving that recognisability implies definability is that there is no obvious way for an automaton to process structures such as graphs, hypergraphs, and matroids, and therefore it is not clear what it means for a class of graphs to be recognisable. On the other hand, a tree decomposition can be processed by an automaton, and in fact, it is not difficult to show that recognisability implies definability for classes of tree decompositions: the proof involves a sentence that asserts the existence of an accepting run, just as in the proof of the BET Theorem. Now transductions come to the rescue. Let $\mathcal{G}$ be a class of graphs with bounded tree-width. Bojańczyk and Pilipczuk [7] prove that there is a transduction from the

monadic language of $\mathcal{G}$ to the monadic language of tree decompositions taking each graph in $\mathcal{G}$ to possible tree decompositions. The Backwards Translation Theorem, applied to this transduction, shows that if the tree decompositions of graphs in $\mathcal{G}$ are recognised by an automaton, then $\mathcal{G}$ is definable. This motivates us to define recognisability of $\mathcal{G}$ to be synonymous with recognisability of its tree decompositions. Under this definition, Bojańczyk and Pilipczuk [7] have proved that recognisability implies definability for minor-closed class of graphs with bounded tree-width, and have thereby settled Courcelle's Conjecture.

I want to conjecture that recognisability implies definability for classes of hypergraphs with bounded decomposition-width. I continue to identify bounded-decomposition-width classes of hypergraphs with the corresponding classes of tree automata. This means that, for me, recognisability of a class of hypergraphs can be defined in terms of a tree automaton that processes tree automata. The following definition is technical, but it attempts to capture the intuition that tree automata can themselves be processed by tree automata.

Let Q be a finite set of states and let s be a non-negative integer. Define $\mathcal{R}(Q,s)$ to be the union of the power sets of $\{\checkmark,\text{✗}\}^s \times \{\checkmark,\text{✗}\}^s \times Q$, Q^3, $\{\checkmark,\text{✗}\}^s \times Q^2$, and $Q \times \{\checkmark,\text{✗}\}^s \times Q$. Thus $\mathcal{R}(Q,s)$ is the set of all possible transition relations. Let Q and Q_1 be finite sets of states, and let $\mu\colon Q \to Q_1$ be an injective function. If $\tau_v \in \mathcal{R}(Q,s)$ is a transition relation, then $\mu(\tau_v) \in \mathcal{R}(Q_1,s)$ is the transition relation obtained by applying μ to every coordinate of every triple in τ_v (we assume that μ acts as the identity on $\{\checkmark,\text{✗}\}^s$). The idea here is that we can use the function μ to rename the states of an automaton.

Let Q_1 and Q_2 be finite sets such that $|Q_1| = k$, and let acc_2 be a subset of Q_2. The set Q_1 is a "canonical" set of k states, and Q_2 is the set of states of the automaton that processes tree automata. We also have a function $\xi\colon \mathcal{R}(Q_1,1) \to \mathcal{R}(Q_2,0)$. If $\mathbf{T} = (T, \mathsf{root}, \{c_v\}_{v\in I(T)}, 1, Q, \mathsf{acc}, \{\tau_v\}_{v\in I(T)})$ is a tree automaton with $|Q| \leq k$, we say that $(Q_1, Q_2, \mathsf{acc}_2, \xi)$ *recognises* $\mathbf{T}$ if there is an injection $\mu\colon Q \to Q_1$ such that $(T, \mathsf{root}, \{c_v\}_{v\in I(T)}, 0, Q_2, \mathsf{acc}_2, \{\xi(\mu(\tau_v))\}_{v\in I(T)})$ is a tree automaton with an accepting run.

Example 6.1 Let $Q_1 = \{0,1\}$ and let $Q_2 = \{Y,N\}$, where $\mathsf{acc}_2 = \{Y\}$. We define ξ so that it takes $\{(\checkmark,\checkmark,0),(\checkmark,\text{✗},1),(\text{✗},\checkmark,1),(\text{✗},\text{✗},0)\}$ to the relation $\{(\epsilon,\epsilon,Y)\}$, and every other subset of $\{\checkmark,\text{✗}\} \times \{\checkmark,\text{✗}\} \times Q_1$ to $\{(\epsilon,\epsilon,N)\}$. (Remember that $\{\checkmark,\text{✗}\}^0 = \{\epsilon\}$.) Similarly, we set ξ to take $\{(1,1,0),(1,0,1),(0,1,1),(0,0,0)\}$ to the relation $\{(Y,Y,Y),(Y,N,N),(N,Y,N),(N,N,N)\}$ and every other subset of Q_1^3 to $\{(Y,Y,N),(Y,N,N),(N,Y,N),(N,N,N)\}$. We will leave it as an exercise to complete the definition of $\xi\colon \mathcal{R}(Q_1,1) \to \mathcal{R}(Q_2,0)$ in such a way that $(Q_1,Q_2,\mathsf{acc}_2,\xi)$ recognises the arity-1 automaton $\mathbf{T}$ if and only if it is deterministic, has at most two states, and accepts exactly the leaf subsets that have even cardinality.

Note that the class of hypergraphs where the hyperedges are exactly the subsets of even cardinality is $\mathrm{CMSO}_1^{\mathrm{hyp}}$-definable, using the modular arithmetic predicates. So in this example, we have a recognisable class that is also definable. The following statement conjectures that recognisability implies definability for general classes of hypergraphs with bounded decomposition-width. Work by Campbell et al. provides progress on the conjecture for finite-field-representable matroids [12].

Conjecture 6.2 *Let $\mathcal{M}$ be a class of hypergraphs with decomposition-width at most k. Assume there is a 4-tuple $(Q_1, Q_2, \mathsf{acc}_2, \xi)$ where Q_1 and Q_2 are finite sets with $|Q_1| = k$, acc_2 is a subset of Q_2, ξ is a function from $\mathcal{R}(Q_1, 1)$ to $\mathcal{R}(Q_2, 0)$, and the following holds for every hypergraph M with decomposition-width at most k:*

(i) *if M is in $\mathcal{M}$, there is a tree automaton $\mathbf{T}$ with at most k states such that $M = (L(\mathbf{T}, \mathcal{I}(\mathbf{T}))$ and $(Q_1, Q_2, \mathsf{acc}_2, \xi)$ recognises $\mathbf{T}$,*

(ii) *if M is not in $\mathcal{M}$ and $\mathbf{T}$ is a tree automaton with at most k states such that $M = (L(\mathbf{T}, \mathcal{I}(\mathbf{T}))$, then $(Q_1, Q_2, \mathsf{acc}_2, \xi)$ does not recognise $\mathbf{T}$.*

Then $\mathcal{M}$ is $\mathrm{CMSO}_1^{\mathrm{hyp}}$-definable.

7 Open problems

In this section I summarise conjectures and questions arising from the survey.

7.1 Complexity

The NP-complete EXACT COVER problem has as its input a hypergraph $(E, \mathcal{I})$. It asks if there is a subcollection $\mathcal{I}' \subseteq \mathcal{I}$ such that each element of E is contained in exactly one member of $\mathcal{I}'$. I conjecture that this problem remains NP-complete even for hypergraphs of bounded decomposition-width. Recall that, for the purposes of describing a hypergraph as input to an algorithm, we think of a tree automaton $\mathbf{T}$ as being a description of $(L(\mathbf{T}), \mathcal{I}(\mathbf{T}))$.

Conjecture 7.1 *There exists an integer k such that* EXACT COVER *is* NP*-complete even when the input is restricted to hypergraphs described by tree automata with at most k states.*

Hypergraphs with exact covers are $\mathrm{CMSO}_2^{\mathrm{hyp}}$-definable, so Conjecture 7.1 would give us reason to believe that we cannot expect an analogue of Courcelle's Conjecture for the $\mathrm{CMSO}_2^{\mathrm{hyp}}$ language.

The DECOMPOSITION-WIDTH problem has as its input an integer k and a hypergraph $M = (E, \mathcal{I})$, described via a list of the members of $\mathcal{I}$. The question is whether M is equal to $(L(\mathbf{T}), \mathcal{I}(\mathbf{T}))$ for some tree automaton $\mathbf{T}$ with at most k states. It is not even clear that the problem is in NP. Given a tree automaton $\mathbf{T}$, we could verify in polynomial-time that $\mathcal{I} \subseteq \mathcal{I}(\mathbf{T})$, but there is no obvious way to certify the reverse containment.

Conjecture 7.2 DECOMPOSITION-WIDTH *is* NP*-hard.*

The next conjecture should be quite simple to resolve.

Conjecture 7.3 *It is* NP*-hard to test $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable properties for Γ-gain-graphic matroids when Γ is infinite, even when the input is restricted to a class with bounded branch-width. It is* NP*-hard to test $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable properties for Γ-gain-graphic matroids when the input has unbounded branch-width, even when Γ is finite.*

7.2 Decidability

Conjecture 7.4 *Let Γ be a finite group and let $\mathcal{M}$ be a minor-closed class of Γ-gain-graphic matroids. The $\mathrm{CMSO}_1^{\mathrm{ind}}$-theory of $\mathcal{M}$ is decidable if and only if $\mathcal{M}$ has bounded branch-width.*

Let Q_n stand for the square grid graph with n^2 vertices. The next conjecture is probably not too difficult to prove, and would help to prove the "only if" direction of Conjecture 7.4.

Conjecture 7.5 *Let $\mathcal{M}$ be a minor-closed class of frame matroids. Assume that for every positive integer n, there is a linear class of cycles $\mathcal{B}_n$ in Q_n such that $F(Q_n, \mathcal{B}_n)$ is isomorphic to a matroid in $\mathcal{M}$. The $\mathrm{CMSO}_1^{\mathrm{ind}}$-theory of $\mathcal{M}$ is undecidable.*

The following open question concerns a general analogue of Seese's Theorem for matroids. I am not optimistic that it will have a clean answer.

Question 7.6 *Which minor-closed classes of matroids have decidable $\mathrm{CMSO}_1^{\mathrm{ind}}$-theories?*

7.3 Definability

I would dearly like an answer to the following question.

Question 7.7 *What are the infinite groups Γ such that the class of Γ-gain-graphic matroids is $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable?*

This is closely related to the question: what are the infinite groups Γ such that the class of Γ-gainable biased graphs is $\mathrm{CMSO}_1^{\mathrm{bias}}$-definable?

The class of representable matroids is not $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable [20]. By analogy, we make the following conjecture.

Conjecture 7.8 *The class of gainable biased graphs is not $\mathrm{CMSO}_1^{\mathrm{bias}}$-definable.*

The next conjecture lists a handful of well-studied classes of matroids. I do not believe any of them can be defined in $\mathrm{CMSO}_1^{\mathrm{ind}}$, but at the same time, I cannot see how to prove this fact using Lemma 5.4.

Conjecture 7.9 *The following classes of matroids are not $\mathrm{CMSO}_1^{\mathrm{ind}}$-definable:* (i) *orientable matroids,* (ii) *transversal matroids,* (iii) *matroids that are transversal and cotransversal,* (iv) *gammoids,* (v) *positroids.*

7.4 Excluded minors and well-quasi-ordering

The work by myself and Funk to build a transduction from frame matroids to their biased graph representations should allow us to attack the following variant of Rota's Conjecture.

Conjecture 7.10 *Let Γ be a finite group. There are only finitely many excluded minors for the class of Γ-gain-graphic matroids.*

The hypothesis of finiteness for Γ is necessary for Conjecture 7.10, because if Γ is infinite then there are infinitely many excluded minors [19, Theorem 2.19].

The class of Γ-gainable biased graphs is minor-closed, so we can ask about its excluded minors in the minor order of biased graphs. Zaslavsky found that there is a single excluded minor when Γ is the group $\mathbb{Z}_2$ [62, Theorem 6], and Brettell, Campbell, Funk, and Mayhew have found the three excluded minors for the case when Γ is $\mathbb{Z}_3$ [10]. Funk shows there are infinitely many excluded minors when Γ is infinite [19, Theorem 2.12]. We make a different analogue of Rota's Conjecture.

Conjecture 7.11 *Let Γ be a finite group. There are only finitely many excluded minors for the class of Γ-gainable biased graphs.*

Despite their similarity, Conjectures 7.10 and 7.11 are probably independent, because a Γ-gain-graphic matroid may be represented by multiple different Γ-gainable biased graphs. For the same reason, the next two conjectures will probably be independent. The first appears as Conjecture 1.4 in [21]. Both conjectures fail if Γ is infinite (Theorem 2.4 and Corollary 2.31 in [19]).

Conjecture 7.12 *Let Γ be a finite group. The class of Γ-gain-graphic matroids is well-quasi-ordered under minors.*

Conjecture 7.13 *Let Γ be a finite group. The class of Γ-gainable biased graphs is well-quasi-ordered under minors.*

Acknowledgements

My thinking on the topics in this survey have been influenced by conversations with many people, including Sapir Ben Shahar, Nathan Bowler, Nick Brettell, Rutger Campbell, Rod Downey, Daryl Funk, Noam Greenberg, Eun Jung Kim, Noleen Koehler, Dugald Macpherson, Janos Makowsky, Angus Matthews, Mike Newman, and Geoff Whittle. Rutger Campbell also provided detailed comments on an earlier draft. I thank them all, as well as an anonymous reviewer for their thoughtful suggestions.

References

[1] René van Bevern, Rodney G. Downey, Michael R. Fellows, Serge Gaspers, and Frances A. Rosamond, *Myhill-Nerode methods for hypergraphs*, Algorithmica **73** (2015), no. 4, 696–729. MR 3432437

[2] Garrett Birkhoff, *Abstract Linear Dependence and Lattices*, Amer. J. Math. **57** (1935), no. 4, 800–804. MR 1507113

[3] ———, *Combinatorial relations in projective geometries*, Ann. of Math. (2) **36** (1935), no. 3, 743–748. MR 1503250

[4] Robert E. Bixby, *On Reid's characterization of the ternary matroids*, J. Combin. Theory Ser. B **26** (1979), no. 2, 174–204. MR 532587

[5] A. Blumensath and B. Courcelle, *Recognizability, hypergraph operations, and logical types*, Inform. and Comput. **204** (2006), no. 6, 853–919. MR 2229541

[6] Mikołaj Bojańczyk, *The category of MSO transductions*, 2023, https://arxiv.org/abs/2305.18039.

[7] Mikołaj Bojańczyk and Michał Pilipczuk, *Definability equals recognizability for graphs of bounded treewidth*, Proceedings of the 31st Annual ACM-IEEE Symposium on Logic in Computer Science (LICS 2016), ACM, New York, 2016, p. 10. MR 3776760

[8] Joseph E. Bonin, *Lattice path matroids: the excluded minors*, J. Combin. Theory Ser. B **100** (2010), no. 6, 585–599. MR 2718679

[9] Nick Brettell, *The excluded minors for* GF(5)*-representable matroids on ten elements*, Adv. in Appl. Math. **166** (2025), 102864.

[10] Nick Brettell, Rutger Campbell, Daryl Funk, and Dillon Mayhew, *The excluded minors for $\mathbb{Z}_3$-gainable and regular biased graphs*, in preparation.

[11] J. Richard Büchi, *Weak second-order arithmetic and finite automata*, Z. Math. Logik Grundlagen Math. **6** (1960), 66–92. MR 125010

[12] Rutger Campbell, Bruno Guillon, Mamadou Moustapha Kanté, Eun Jung Kim, and Sang-il Oum, *Recognisability equals definability for finitely representable matroids of bounded path-width*, 2025 40th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), 2025, pp. 678–690.

[13] Bruno Courcelle, *The monadic second-order logic of graphs. I. Recognizable sets of finite graphs*, Inform. and Comput. **85** (1990), no. 1, 12–75. MR 1042649

[14] ———, *The monadic second-order logic of graphs. V. On closing the gap between definability and recognizability*, Symposium on Mathematical Foundations of Computer Science '89 (Porabka-Kozubnik, 1989), vol. 80, 1991, pp. 153–202. MR 1117063

[15] Bruno Courcelle and Joost Engelfriet, *Graph structure and monadic second-order logic*, Encyclopedia of Mathematics and its Applications, vol. 138, Cambridge University Press, Cambridge, 2012, A language-theoretic approach, With a foreword by Maurice Nivat. MR 2962260

[16] R. G. Downey and M. R. Fellows, *Parameterized complexity*, Monographs in Computer Science, Springer-Verlag, New York, 1999. MR 1656112

[17] Heinz-Dieter Ebbinghaus and Jörg Flum, *Finite model theory*, Perspectives in Mathematical Logic, Springer-Verlag, Berlin, 1995. MR 1409813

[18] Calvin C. Elgot, *Decision problems of finite automata design and related arithmetics*, Trans. Amer. Math. Soc. **98** (1961), 21–51. MR 139530

[19] Daryl Funk, *On excluded minors and biased graph representations of frame matroids*, Ph.D. thesis, Simon Fraser University, 2015.

[20] Daryl Funk, Angus Matthews, and Dillon Mayhew, *Monadic non-definability and gain-graphic matroids*, 2025, https://arxiv.org/abs/2510.00139.

[21] Daryl Funk, Dillon Mayhew, and Mike Newman, *Defining bicircular matroids in monadic logic*, Q. J. Math. **73** (2022), no. 1, 65–92. MR 4395073

[22] _____, *Tree automata and pigeonhole classes of matroids: I*, Algorithmica **84** (2022), no. 7, 1795–1834. MR 4444152

[23] _____, *Tree automata and pigeonhole classes of matroids: II*, Electron. J. Combin. **30** (2023), no. 3, Paper No. 3.6, 34. MR 4614540

[24] J. F. Geelen, A. M. H. Gerards, and A. Kapoor, *The excluded minors for* GF(4)*-representable matroids*, J. Combin. Theory Ser. B **79** (2000), no. 2, 247–299. MR 1769191

[25] Jim Geelen, Bert Gerards, and Geoff Whittle, *Towards a structure theory for matrices and matroids*, International Congress of Mathematicians. Vol. III, Eur. Math. Soc., Zürich, 2006, pp. 827–842. MR 2275708

[26] _____, *Excluding a planar graph from* GF(q)*-representable matroids*, J. Combin. Theory Ser. B **97** (2007), no. 6, 971–998. MR 2354713

[27] _____, *Towards a matroid-minor structure theory*, Combinatorics, complexity, and chance, Oxford Lecture Ser. Math. Appl., vol. 34, Oxford Univ. Press, Oxford, 2007, pp. 72–82. MR 2314562

[28] _____, *Structure in minor-closed classes of matroids*, Surveys in combinatorics 2013, London Math. Soc. Lecture Note Ser., vol. 409, Cambridge Univ. Press, Cambridge, 2013, pp. 327–362. MR 3184116

[29] _____, *Solving Rota's conjecture*, Notices Amer. Math. Soc. **61** (2014), no. 7, 736–743. MR 3221124

[30] Petr Hliněný, *On matroid representability and minor problems*, Mathematical Foundations of Computer Science 2006 (Berlin, Heidelberg) (Rastislav Královič and Paweł Urzyczyn, eds.), Springer Berlin Heidelberg, 2006, pp. 505–516.

[31] Petr Hliněný, *Branch-width, parse trees, and monadic second-order logic for matroids (extended abstract)*, STACS 2003, Lecture Notes in Comput. Sci., vol. 2607, Springer, Berlin, 2003, pp. 319–330. MR 2066603

[32] _____, *On matroid properties definable in the MSO logic*, Mathematical foundations of computer science 2003, Lecture Notes in Comput. Sci., vol. 2747, Springer, Berlin, 2003, pp. 470–479. MR 2081597

[33] Petr Hliněný and Detlef Seese, *Trees, grids, and MSO decidability: from graphs to matroids*, Theoret. Comput. Sci. **351** (2006), no. 3, 372–393. MR 2202497

[34] Susan Jowett, Dillon Mayhew, Songbao Mo, and Christopher Tuffley, *Monadic transductions and definable classes of matroids*, 2024, https://arxiv.org/abs/2401.12969.

[35] J. Kahn and J. P. S. Kung, *Varieties of combinatorial geometries*, Trans. Amer. Math. Soc. **271** (1982), no. 2, 485–499. MR 654846

[36] Richard M. Karp, *Reducibility among combinatorial problems*, Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972), The IBM Research Symposia Series, Plenum, New York-London, 1972, pp. 85–103. MR 378476

[37] Tomer Kotek and Johann A. Makowsky, *Connection matrices and the definability of graph parameters*, Log. Methods Comput. Sci. **10** (2014), no. 4, 4:1, 33. MR 3274646

[38] Daniel Král, *Decomposition width of matroids*, Discrete Appl. Math. **160** (2012), no. 6, 913–923. MR 2901097

[39] Leonid Libkin, *Elements of finite model theory*, Texts in Theoretical Computer Science. An EATCS Series, Springer-Verlag, Berlin, 2004. MR 2102513

[40] Saunders Mac Lane, *Some Interpretations of Abstract Linear Dependence in Terms of Projective Geometry*, Amer. J. Math. **58** (1936), no. 1, 236–240. MR 1507146

[41] Dillon Mayhew, *Matroid complexity and nonsuccinct descriptions*, SIAM J. Discrete Math. **22** (2008), no. 2, 455–466. MR 2399359

[42] Dillon Mayhew, Mike Newman, and Geoff Whittle, *Yes, the 'missing axiom' of matroid theory is lost forever*, Trans. Amer. Math. Soc. **370** (2018), no. 8, 5907–5929. MR 3803151

[43] John Myhill, *Finite automata and the representation of events*, WADD Technical Report **57** (1957), 112–137.

[44] Takeo Nakasawa, *Zur Axiomatik der linearen Abhängigkeit. I [Sci. Rep. Tokyo Bunrika Daigaku Sect. A* **2** *(1935), no. 43, 129–149; Zbl 0012.22001]*, A lost mathematician, Takeo Nakasawa, Birkhäuser, Basel, 2009, pp. 68–88. MR 2516552

[45] A. Nerode, *Linear automaton transformations*, Proc. Amer. Math. Soc. **9** (1958), 541–544. MR 135681

[46] Sang-il Oum and Paul Seymour, *Approximating clique-width and branch-width*, J. Combin. Theory Ser. B **96** (2006), no. 4, 514–528. MR 2232389

[47] James Oxley, *Matroid theory*, second ed., Oxford Graduate Texts in Mathematics, vol. 21, Oxford University Press, Oxford, 2011. MR 2849819

[48] M. O. Rabin and D. Scott, *Finite automata and their decision problems*, IBM J. Res. Develop. **3** (1959), 114–125. MR 103795

[49] Neil Robertson and P. D. Seymour, *Graph minors. V. Excluding a planar graph*, J. Combin. Theory Ser. B **41** (1986), no. 1, 92–114. MR 854606

[50] ———, *Graph minors. XX. Wagner's conjecture*, J. Combin. Theory Ser. B **92** (2004), no. 2, 325–357. MR 2099147

[51] Gian-Carlo Rota, *Combinatorial theory, old and new*, Actes du Congrès International des Mathématiciens (Nice, 1970), Tome 3, Gauthier-Villars Éditeur, Paris, 1971, pp. 229–233. MR 505646

[52] D. Seese, *The structure of the models of decidable monadic theories of graphs*, Ann. Pure Appl. Logic **53** (1991), no. 2, 169–195. MR 1114848

[53] P. D. Seymour, *Matroid representation over* GF(3), J. Combin. Theory Ser. B **26** (1979), no. 2, 159–173. MR 532586

[54] Ernst Steinitz, *Algebraische Theorie der Körper*, J. Reine Angew. Math. **137** (1910), 167–309. MR 1580791

[55] Yann Strozecki, *Monadic second-order model-checking on decomposable matroids*, Discrete Appl. Math. **159** (2011), no. 10, 1022–1039. MR 2794251

[56] B. A. Trahtenbrot, *Finite automata and the logic of one-place predicates*, Sibirsk. Mat. Ž. **3** (1962), 103–131. MR 147392

[57] W. T. Tutte, *A homotopy theorem for matroids. I, II*, Trans. Amer. Math. Soc. **88** (1958), 144–174. MR 101526

[58] William Thomas Tutte, *Selected papers of W. T. Tutte. Vol. II*, Charles Babbage Research Centre, Winnipeg, MB, 1979. MR 526767

[59] P. Vámos, *The missing axiom of matroid theory is lost forever*, J. London Math. Soc. (2) **18** (1978), no. 3, 403–408. MR 518224

[60] Bartel Leendert van der Waerden, *Moderne Algebra*, J. Springer, Berlin, 1940. MR 2841

[61] Hassler Whitney, *On the Abstract Properties of Linear Dependence*, Amer. J. Math. **57** (1935), no. 3, 509–533. MR 1507091

[62] Thomas Zaslavsky, *Characterizations of signed graphs*, J. Graph Theory **5** (1981), no. 4, 401–406. MR 635702

[63] ———, *Biased graphs. I. Bias, balance, and gains*, J. Combin. Theory Ser. B **47** (1989), no. 1, 32–52. MR 1007712

[64] ———, *Biased graphs. II. The three matroids*, J. Combin. Theory Ser. B **51** (1991), no. 1, 46–72. MR 1088626

School of Computer Science
University of Leeds
Leeds
United Kingdom
d.mayhew@leeds.ac.uk