Logical Methods in Computer Science
Volume 19, Issue 3, 2023, pp. 6:1–6:13
https://lmcs.episciences.org/

Submitted Jul. 20, 2022
Published Jul. 14, 2023

CAPTURING THE POLYNOMIAL HIERARCHY BY SECOND-ORDER REVISED KROM LOGIC

KEXU WANG [a,*], SHIGUANG FENG [b,*], AND XISHUN ZHAO [a]

[a] Institute of Logic and Cognition, Sun Yat-sen University, Guangzhou, 510275, China

[b] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, 510006, China
e-mail address: fengshg3@mail.sysu.edu.cn

Abstract. We study the expressive power and complexity of second-order revised Krom logic (SO-KROM^r). On ordered finite structures, we show that its existential fragment Σ^1_1-KROM^r equals Σ^1_1-KROM, and captures NL. On all finite structures, for $k \geq 1$, we show that Σ^1_k equals Σ^1_{k+1}-KROM^r if k is even, and Π^1_k equals Π^1_{k+1}-KROM^r if k is odd. The results give an alternative logic to capture the polynomial hierarchy. We also introduce an extended version of second-order Krom logic (SO-EKROM). On ordered finite structures, we prove that SO-EKROM collapses to Π^1_2-EKROM and equals Π^1_1. Both SO-EKROM and Π^1_2-EKROM capture co-NP on ordered finite structures.

Introduction

Descriptive complexity studies the logical characterization of computational complexity classes. It describes the property of a problem using the logical method. Computational complexity considers the computational resources such as time and space needed to decide a problem, whereas descriptive complexity explores the minimal logic that captures a complexity class. We say that a logic $\mathcal{L}$ captures a complexity class $\mathcal{C}$, if (i) the data complexity of $\mathcal{L}$ is in $\mathcal{C}$, i.e., for every $\mathcal{L}$ formula φ, the set of models of φ is decidable in $\mathcal{C}$; and (ii) if a class of finite structures is in $\mathcal{C}$, then it is definable by an $\mathcal{L}$ formula. Moreover, if two logics $\mathcal{L}_1$ and $\mathcal{L}_2$ capture two complexity classes $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively, then $\mathcal{L}_1$ and $\mathcal{L}_2$ have the same expressive power if and only if $\mathcal{C}_1$ is equal to $\mathcal{C}_2$ [EF95]. So the equivalence problem between different complexity classes can be transformed into the expressive power problem of different logics. In 1974, Fagin showed that the existential fragment of second-order logic ($\exists$SO) captures NP [Fag74]. This seminal work had been followed by many studies in the logical characterization of complexity classes. In 1982, Immerman and Vardi independently showed that the least fixed-point logic FO(LFP) captures P on ordered finite structures [Imm82, Var82]. In 1987, Immerman showed that the deterministic transitive closure logic FO(DTC) and transitive closure logic FO(TC) capture L and NL on ordered

Key words and phrases: polynomial hierarchy, Krom logic, descriptive complexity, second-order logic.
Corresponding author: Shiguang Feng.
* These authors contributed equally to this work.

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.46298/LMCS-19(3:6)2023

finite structures, respectively [Imm87]. In 1989, Abiteboul and Vianu showed that the partial fixed-point logic FO(PFP) captures PSPACE on ordered finite structures [AV89].

Whether P equals NP is an important problem in theoretical computer science. ∃SO captures NP on all finite structures. Hence, no logic capturing P on all finite structures implies P $\neq$ NP. The capturing result of FO(LFP) for P is on ordered structures. Actually, FO(LFP) even cannot express the parity of a structure [EF95]. So finding a logic that can capture P effectively on all finite structures is of importance. Many extensions of FO(LFP) had been studied. FO(IFP, #) is obtained by adding counting quantifiers to the inflationary fixed-point logic FO(IFP) which has the same expressive power as FO(LFP) [GO92, Ott96]. FO(IFP,rank) is an extension of FO(IFP) with the rank operator that can define the rank of a matrix [Daw08, DGHL09, ABD09]. Both FO(IFP, #) and FO(IFP,rank) are strictly more expressive than FO(LFP), but neither of them captures P on all finite structures [EF95, DGP19]. Second-order logic and its fragments are further candidates of logics for P. In [Grä91], Grädel showed that SO-HORN captures P on ordered finite structures. Feng and Zhao introduced second-order revised Horn logic (SO-HORN^r) and showed that it equals FO(LFP) on all finite structures [FZ12, FZ13].

Similar to the results for P, it is easy to check that no logic capturing NL on all finite structures implies NL $\neq$ NP. Grädel showed that SO-KROM captures NL on ordered finite structures [Grä92]. Cook and Kolokolova introduced the second-order theory V-Krom of bounded arithmetic for NL that is based on SO-KROM [CK04]. In this paper, we introduce second-order revised Krom logic (SO-KROM^r). It is an extension of SO-KROM by allowing the formula $\exists\bar{z}R\bar{z}$ in the clauses where R is a second-order variable. SO-KROM^r is strictly more expressive than SO-KROM. Its existential fragment Σ^1_1-KROM^r is equivalent to SO-KROM on ordered finite structures. For all $k \geq 1$, on all finite structures, we show that every Σ^1_k formula is equivalent to a Σ^1_{k+1}-KROM^r formula for even k, and every Π^1_k formula is equivalent to a Π^1_{k+1}-KROM^r formula for odd k. Hence, every second-order formula is equivalent to an SO-KROM^r formula. For the data complexity of SO-KROM^r, we show that Σ^1_{k+1}-KROM^r is in Σ^p_k for even k, and Π^1_{k+1}-KROM^r is in Π^p_k for odd k, where $\Sigma^p_0 = \Pi^p_0 = \mathrm{P}$, Σ^p_{k+1} is the set of decision problems solvable in nondeterministic polynomial time by a Turing machine augmented with an oracle in Σ^p_k, and Π^p_{k+1} is the complement of Σ^p_{k+1} [Sto76]. The polynomial time hierarchy $\mathrm{PH} = \bigcup_{k=0}^{\infty} \Sigma^p_k$, which is contained within PSPACE. It is well-known that the second-order formulas Σ^1_k (resp., Π^1_k) capture Σ^p_k (resp., Π^p_k) ($k \geq 1$) [Imm98]. Combining these we see that SO-KROM^r gives an alternative logical characterization for PH, which is an interesting result in the field of descriptive complexity. The main results in the paper are summarized in Figure 1.

The paper is organized as follows. In Section 1, we give the basic definitions and notations. In Section 2, we study the expressive power and complexity of the existential fragment of SO-KROM^r. In Section 3, we study the descriptive complexity of SO-KROM^r. In Section 4, we introduce second-order extended Krom logic and study its descriptive complexity. Section 5 is the conclusion of the paper.

1. Preliminaries

Let $\tau = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m, P_1, P_2, \ldots, P_n\}$ be a vocabulary, where $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_m$ are constant symbols and $P_1, P_2, \ldots, P_n$ are relation symbols. A *τ-structure* $\mathcal{A}$ is a tuple

$$\langle A, \mathbf{c}_1^A, \mathbf{c}_2^A, \ldots, \mathbf{c}_m^A, P_1^A, P_2^A, \ldots, P_n^A \rangle,$$

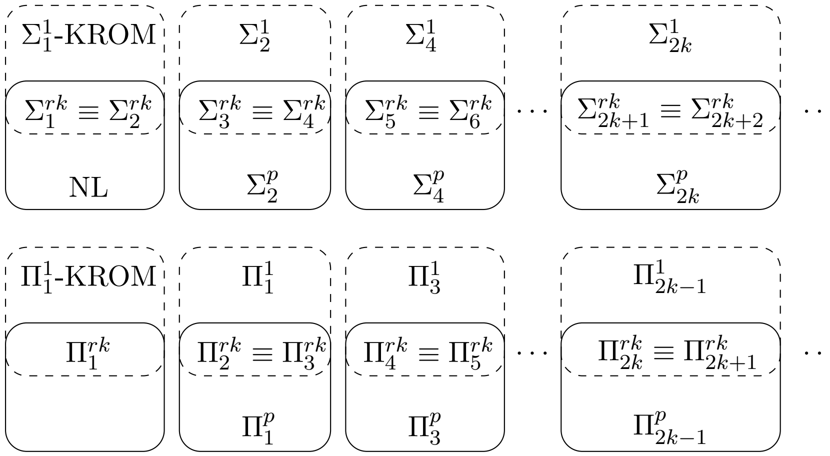

Figure 1: The expressive power and complexity of SO-KROM^r. Σ^{rk}_k and Π^{rk}_k denote Σ^1_k-KROM^r and Π^1_k-KROM^r, respectively. The dashed rectangle parts show the equivalence relation between second-order formulas and SO-KROM^r formulas. The solid rectangle parts show the capturing results of SO-KROM^r for PH.

where A is the domain of $\mathcal{A}$, and $\mathbf{c}^A_1, \mathbf{c}^A_2, \ldots, \mathbf{c}^A_m$, $P^A_1, P^A_2, \ldots, P^A_n$ are the interpretations of the constant and relation symbols over A, respectively. We assume the identity relation "=" is contained in every vocabulary, and omit the superscript "A" in the notation when no confusion is caused. We call $\mathcal{A}$ finite if its domain A is a (nonempty) finite set. In this paper, all structures considered are finite. We use $|\ |$ to denote the cardinality of a set or the arity of a tuple, e.g., $|\{a, b, c\}| = 3$ and $|\bar{x}| = 3$ where $\bar{x} = (x_1, x_2, x_3)$, and $arity(X)$ to denote the arity of a relation symbol (variable) X. A finite structure is *ordered* if it is equipped with a linear order relation "$\leq$", a successor relation "SUCC", and constants "**min**" and "**max**" interpreted as the minimal and maximal elements, respectively.

Given a logic $\mathcal{L}$, we use $\mathcal{L}(\tau)$ to denote the set of $\mathcal{L}$ formulas over vocabulary τ. For better readability, the symbol "τ" is omitted when it is clear from context. Given two logics $\mathcal{L}_1$ and $\mathcal{L}_2$, we use $\mathcal{L}_1 \leq \mathcal{L}_2$ to denote that every $\mathcal{L}_1$ formula is equivalent to an $\mathcal{L}_2$ formula. If both $\mathcal{L}_1 \leq \mathcal{L}_2$ and $\mathcal{L}_2 \leq \mathcal{L}_1$ hold, then we write $\mathcal{L}_1 \equiv \mathcal{L}_2$.

Definition 1.1. Given a vocabulary τ, the second-order Krom logic over τ, denoted by SO-KROM(τ), is a set of second-order formulas of the form

$$Q_1 R_1 \cdots Q_m R_m \forall \bar{x}(C_1 \wedge \cdots \wedge C_n)$$

where each $Q_i \in \{\forall, \exists\}$, $C_1, \ldots, C_n$ are Krom clauses with respect to $R_1, \ldots, R_m$, more precisely, each C_j is a disjunction of the form

$$\beta_1 \vee \cdots \vee \beta_q \vee H_1 \vee H_2,$$

where

(1) each β_s for $s \in \{1, \ldots, q\}$ is either $P\bar{y}$ or $\neg P\bar{y}$ ($P \in \tau$);
(2) each H_t is either $R_i\,\bar{z}$, $\neg R_i\,\bar{z}$ ($1 \leq i \leq m$), or $\perp$ (for false).

If we replace (2) by

(2′) each H_t is either $R_i\,\bar{z}$, $\neg R_i\,\bar{z}$, $\exists z_1 \cdots \exists z_{arity(R_i)}\, R_i\, z_1 \ldots z_{arity(R_i)}$ ($1 \leq i \leq m$), or $\perp$ (for false), then we call this logic second-order revised Krom Logic, denoted by SO-KROM$^r(\tau)$.

We use Σ^1_k-KROM^r (resp., Π^1_k-KROM^r) to denote the set of SO-KROM^r formulas whose second-order prefix starts with an existential (resp., a universal) quantifier and alternates $k-1$ times between series of existential and universal quantifiers.

Example 1.2. A directed graph is strongly connected iff there exists a path between every pair of nodes. The strong connectivity problem is NL-complete, which is defined by

Input: a directed graph $G = (V, E)$,
Output: yes if G is strongly connected, and no otherwise.

Since NL = co-NL, the complement of the strong connectivity problem is also NL-complete, which can be defined by the following Σ^1_1-KROM^r formula

$$\exists R \exists Y \forall x \forall y \forall z \left(\begin{array}{l} (Exy \to Rxy) \wedge (Exy \wedge Ryz \to Rxz) \\ \wedge(\neg Rxy \leftrightarrow Yxy) \wedge \exists u \exists v Yuv \end{array} \right)$$

where R is the transitive closure of E, and Y is the complement of R. A graph G satisfies the formula iff there exist two nodes a, b such that a cannot reach b.

SO-KROM is closed under substructures [Grä92], which means that if a structure satisfies a SO-KROM formula then all its substructures also satisfy the formula. Because a non-strongly connected graph may be made strongly connected by removing nodes, SO-KROM cannot define the complement of the strong connectivity problem. The above example shows that SO-KROM^r is strictly more expressive than SO-KROM.

2. The expressive power and complexity of Σ^1_1-KROM^r

In this section, we study the expressive power of the universal and existential fragments of SO-KROM^r, and show that Σ^1_1-KROM^r captures NL on finite ordered structures.

Proposition 2.1. *Every Π^1_1-KROM^r formula is equivalent to a first-order formula $\forall \bar{x}\varphi$, where φ is a quantifier-free CNF formula.*

Proof. Given a Π^1_1-KROM^r formula $\Phi = \forall X_1 \dots \forall X_n \forall \bar{x}(C_1 \wedge \cdots \wedge C_m)$, we deal with each clause C_j for $j \in \{1, \dots, m\}$ as follows. Let α denote the first-order part of C_j.

Case 1: If $C_j = \alpha \vee \neg X_i\bar{x}_1 \vee \exists \bar{x}_2 X_i \bar{x}_2$, then remove the clause C_j.
Case 2: If $C_j = \alpha \vee X_i\bar{x}_1 \vee \neg X_i\bar{x}_2$, then replace C_j by $\alpha \vee \bar{x}_1 = \bar{x}_2$.
Case 3: For the other cases, remove all occurrences of second-order variables in C_j.

After the above steps, all second-order variables are removed and we obtain a first-order formula $\phi = \forall \bar{x}(C'_1 \wedge \cdots \wedge C'_{m'})$, where each C'_j is quantifier-free.

In **Case 1**, $\neg X_i\bar{x}_1 \vee \exists \bar{x}_2 X_i\bar{x}_2$ is a tautology, so the clause can be removed safely. In **Case 2**, if $\bar{x}_1 = \bar{x}_2$, then $X_i\bar{x}_1 \vee \neg X_i\bar{x}_2$ is always true; if $\bar{x}_1 \neq \bar{x}_2$, then there exists a valuation for X_i such that $X_i\bar{x}_1 \vee \neg X_i\bar{x}_2$ is false, the clause is true iff α is true. Hence, $\forall X_i \forall \bar{x}(\alpha \vee X_i\bar{x}_1 \vee \neg X_i\bar{x}_2)$ is equivalent to $\forall \bar{x}(\alpha \vee \bar{x}_1 = \bar{x}_2)$. In **Case 3**, there is always a valuation for the second-order variables in C_j under which the second-order part of C_j is false. So all occurrences of second-order variables can be removed from C_j. Therefore, Φ and ϕ are equivalent. □

Corollary 2.2. *For each $k \geq 1$, if k is odd, then Σ^1_k-KROM$^r \equiv \Sigma^1_{k+1}$-KROM^r; and if k is even, then Π^1_k-KROM$^r \equiv \Pi^1_{k+1}$-KROM^r.*

Proof. If the type of the innermost second-order quantifier block of a SO-KROM^r formula is universal, then it can be removed by Proposition 2.1 to get an equivalent formula. □

We use $\varphi[\alpha/\beta]$ to denote replacing the variable (or the formula) α in φ with β.

Lemma 2.3. *Let $\exists x_1 \dots \exists x_n \phi$ be a quantified Boolean formula. It is equivalent to the following formula*

$$\phi[x_1/\bot, \dots, x_n/\bot] \vee \bigvee_{1 \leq i \leq n} \exists x_1 \dots \exists x_{i-1} \exists x_{i+1} \dots \exists x_n \phi[x_i/\top].$$

Proof. $\exists x_1 \dots \exists x_n \phi$ is true iff ϕ is true when all $x_1, \dots, x_n$ are false, or for some x_i, where $(1 \leq i \leq n)$, the formula $\exists x_1 \dots \exists x_{i-1} \exists x_{i+1} \dots \exists x_n \phi$ is true when x_i is true. □

From Lemma 2.3 we can infer the following proposition.

Proposition 2.4. *Every Σ^1_1-KROM^r formula is equivalent to a formula of the form $\exists \bar{y}_1 \phi_1 \vee \dots \vee \exists \bar{y}_n \phi_n$, where each ϕ_i for $i \in \{1, \dots, n\}$ is a Σ^1_1-KROM formula.*

Proof. Let $\Psi = \exists R \exists \overline{Y} \forall \bar{x} \phi$ be a Σ^1_1-KROM^r formula, and $\alpha(\bar{z}) = (\bar{z} = \bar{y}) \vee R\bar{z}$ where $\bar{y}$ have no occurrence in ϕ. It is easily seen that if $\bar{z} = \bar{y}$ holds then $\alpha(\bar{z})$ is true, and if $\bar{z} \neq \bar{y}$ holds then $\alpha(\bar{z})$ is equivalent to $R\bar{z}$. So $\alpha(\bar{z})$ is equivalent to $R\bar{z}$ except at the point $\bar{y}$. Define

$$\Psi' = \exists \overline{Y} \forall \bar{x} \phi[R\bar{z}/\bot] \vee \exists \bar{y} \exists R \exists \overline{Y} \forall \bar{x} \phi[R\bar{z}/\alpha(\bar{z})].$$

We show that Ψ and Ψ' are equivalent. It is easily seen that for any structure $\mathcal{A}$, $\mathcal{A} \models \Psi$ iff $(\mathcal{A}, R) \models \exists \overline{Y} \forall \bar{x} \phi$, where either $R = \emptyset$ or R is not empty. Every occurrence of $\exists \bar{z} R\bar{z}$ in Ψ' is either replaced by $\exists \bar{z} \bot$ or replaced by $\exists \bar{z} \alpha(\bar{z})$ which is a tautology. We remove the occurrences of $\exists \bar{z} \bot$ and the clauses containing $\exists \bar{z} \alpha(\bar{z})$ in Ψ'. For any structure $\mathcal{A}$, we can construct a quantified Boolean formula $\Psi_\mathcal{A}$ such that $\mathcal{A} \models \Psi$ iff $\Psi_\mathcal{A}$ is true (see the proof of Proposition 3.3 for details of the construction). Similarly, we can construct $\Psi'_\mathcal{A}$ such that $\mathcal{A} \models \Psi'$ iff $\Psi'_\mathcal{A}$ is true. By Lemma 2.3, $\Psi_\mathcal{A}$ and $\Psi'_\mathcal{A}$ are equivalent. Therefore, Ψ and Ψ' are equivalent. The same procedure can be repeated for each $Y_i \in \overline{Y}$ until all occurrences of $\exists \bar{v} Y_i \bar{v}$ are removed. Finally, we can obtain an equivalent formula of the form $\exists \bar{y}_1 \phi_1 \vee \dots \vee \exists \bar{y}_n \phi_n$, where each ϕ_i $(1 \leq i \leq n)$ is a Σ^1_1-KROM formula. □

Proposition 2.5. *The data complexity of Σ^1_1-KROM^r is in* NL.

Proof. By Proposition 2.4, we only need to show that the data complexity of the formula $\exists \bar{y}_1 \phi_1 \vee \dots \vee \exists \bar{y}_n \phi_n$, where each ϕ_i $(1 \leq i \leq n)$ is a Σ^1_1-KROM formula, is in NL. Given a structure $\mathcal{A}$, the Turing machine can nondeterministically choose an $i \in \{1, \dots, n\}$ and a tuple $\bar{u}_i \in A^{|\bar{y}_i|}$ in logarithmic space. Whether $\mathcal{A} \models \phi_i[\bar{u}_i]$ holds can be checked in NL since the data complexity of Σ^1_1-KROM is in NL [Grä92]. □

Every Σ^1_1-KROM formula is also a Σ^1_1-KROM^r formula. Because Σ^1_1-KROM captures NL on ordered finite structures [Grä92], combining Corollary 2.2 we obtain the following corollary.

Corollary 2.6. *Both Σ^1_1-KROM^r and Σ^1_2-KROM^r capture* NL *on ordered finite structures.*

3. The descriptive complexity of SO-KROM^r

SO-KROM collapses to its existential fragment. This is unlikely to be true for SO-KROM^r by the following result. Let Σ_k-CNF (resp., Σ_k-DNF) denote the set of quantified Boolean formulas $\exists\bar{x}_1\forall\bar{x}_2\exists\bar{x}_3\ldots Q_k\bar{x}_k\phi$ whose prefix starts with an existential quantifier and has $k-1$ alternations between series of existential and universal quantifiers, and the matrix ϕ is a quantifier-free formula in conjunctive normal form (resp., disjunctive normal form). The definitions for Π_k-CNF and Π_k-DNF are similar where the formula's prefix starts with a universal quantifier. Given a set $\mathbf{F}$ of quantified Boolean formulas, the evaluation problem of $\mathbf{F}$ is deciding the truth value of the formulas in it. For the polynomial hierarchy, it is shown that the evaluation problem of Σ_k-CNF (resp., Σ_k-DNF) is Σ_k^p-complete if k is odd (resp., even) [Sto76]. Hence, the evaluation problem of Π_k-DNF (resp., Π_k-CNF) is Π_k^p-complete if k is odd (resp., even) by duality.

Proposition 3.1. *The evaluation problem of* Σ_k-DNF *is definable in* Σ^1_{k+1}-KROM^r *if k is even, and the evaluation problem of* Π_k-DNF *is definable in* Π^1_{k+1}-KROM^r *if k is odd.*

Proof. We only prove for Σ_k-DNF where k is even, the proof for Π_k-DNF where k is odd is the same as it. Let vocabulary $\tau = \{\mathrm{Clause}, \mathrm{Var}_1, \ldots, \mathrm{Var}_k, \mathrm{Pos}, \mathrm{Neg}\}$, where Clause, $\mathrm{Var}_1, \ldots, \mathrm{Var}_k$ are unary relation symbols, and Pos, Neg are binary relation symbols. Using a similar method as in [Imm98], we can encode a Σ_k-DNF formula $\exists\bar{x}_1\forall\bar{x}_2\cdots\exists\bar{x}_{k-1}\forall\bar{x}_k\phi$ via a τ-structure $\mathcal{A}$ such that for any $i, j \in A$, $\mathrm{Clause}\, i$ holds iff i is a clause, $\mathrm{Var}_h\, j$ holds iff j is a variable occurring in the quantifier block $\exists(\forall)\bar{x}_h$ for $h \in \{1, \ldots, k\}$, and $\mathrm{Pos}\, ij$ (resp., $\mathrm{Neg}\, ij$) holds iff variable j occurs positively (resp., negatively) in clause i. For example, the Σ_4-DNF formula

$$\exists x_1\forall x_2\exists x_3\forall x_4(\underbrace{(x_1 \wedge \neg x_2)}_{1} \vee \underbrace{(x_2 \wedge \neg x_4)}_{2} \vee \underbrace{(x_3 \wedge x_4)}_{3})$$

can be encoded via the structure $\langle\{1,2,3,4\}, \mathrm{Clause}, \mathrm{Var}_1, \mathrm{Var}_2, \mathrm{Var}_3, \mathrm{Var}_4, \mathrm{Pos}, \mathrm{Neg}\rangle$, where $\mathrm{Clause} = \{1,2,3\}$, $\mathrm{Var}_1 = \{1\}$, $\mathrm{Var}_2 = \{2\}$, $\mathrm{Var}_3 = \{3\}$, $\mathrm{Var}_4 = \{4\}$, $\mathrm{Neg} = \{(1,2),(2,4)\}$, $\mathrm{Pos} = \{(1,1),(2,2),(3,3),(3,4)\}$. Let Φ be the following formula

$$\exists X_1\forall X_2\ldots\exists X_{k-1}\forall X_k\exists Y\forall x\forall y\left(\begin{array}{l}\exists zYz \wedge (Yx \rightarrow \mathrm{Clause}\,x)\wedge \\ \bigwedge_{1\le h\le k}(Yx \wedge \mathrm{Pos}\,xy \wedge \mathrm{Var}_h\,y \rightarrow X_h\,y)\wedge \\ \bigwedge_{1\le h\le k}(Yx \wedge \mathrm{Neg}\,xy \wedge \mathrm{Var}_h\,y \rightarrow \neg X_h\,y)\end{array}\right).$$

Obviously, Φ is a Σ^1_{k+1}-KROM^r formula, and it expresses that there is a valuation X_1 to $\bar{x}_1$, for any valuation X_2 to $\bar{x}_2, \ldots$, there is a valuation X_{k-1} to $\bar{x}_{k-1}$, for any valuation X_k to $\bar{x}_k$, there is a nonempty set Y of clauses such that every literal in the clauses in Y is true under the valuation. For an arbitrary Σ_k-DNF formula ψ, let $\mathcal{A}$ be the τ-structure that encodes ψ, it is easily seen that $\mathcal{A} \models \Phi$ iff ψ is true. □

Before showing that every second-order formula is equivalent to a SO-KROM^r formula, we first prove a lemma.

Lemma 3.2. *Every first-order formula is equivalent to a second-order formula* $\exists Y\forall\bar{x}(\exists\bar{y}Y\bar{z}\bar{y} \wedge C_1 \wedge \cdots \wedge C_m)$ *where each C_i is a disjunction of atomic or negated atomic formulas.*

Proof. Given a first-order formula φ, without loss of generality, assume that φ is in the prenex normal form $\forall\bar{x}_1\exists\bar{y}_1\cdots\forall\bar{x}_n\exists\bar{y}_n(C_1 \wedge \cdots \wedge C_m)$, where each C_i for $i \in \{1, \ldots, m\}$ is a disjunction of atomic or negated atomic formulas. Define

$$\varphi_1 = \forall \bar{x}_1 \cdots \forall \bar{x}_n \exists \bar{y}_1 \cdots \exists \bar{y}_n Y \bar{x}_1 \cdots \bar{x}_n \bar{y}_1 \cdots \bar{y}_n,$$

$$\begin{aligned}\varphi_2 = \; & \forall \bar{x}_1 \cdots \forall \bar{x}_n \forall \bar{y}_1 \cdots \forall \bar{y}_n \forall \bar{x}'_1 \cdots \forall \bar{x}'_n \forall \bar{y}'_1 \cdots \forall \bar{y}'_n \\ & \Big(Y \bar{x}_1 \cdots \bar{x}_n \bar{y}_1 \cdots \bar{y}_n \wedge Y \bar{x}'_1 \cdots \bar{x}'_n \bar{y}'_1 \cdots \bar{y}'_n \\ & \rightarrow \textstyle\bigwedge_{1 \leq i \leq n} \Big((\bigwedge_{1 \leq j \leq i} \bar{x}_j = \bar{x}'_j) \rightarrow \bar{y}_i = \bar{y}'_i \Big) \Big),\end{aligned}$$

$$\varphi_3 = \forall \bar{x}_1 \forall \bar{y}_1 \cdots \forall \bar{x}_n \forall \bar{y}_n (Y \bar{x}_1 \cdots \bar{x}_n \bar{y}_1 \cdots \bar{y}_n \rightarrow \textstyle\bigwedge_{1 \leq i \leq m} C_i).$$

The relation Y encodes a Skolem function for each $\bar{y}_i$ ($1 \leq i \leq n$), whose value only depends on the values of $\bar{x}_1, \ldots, \bar{x}_i$. It is easy to check that φ is equivalent to the formula $\exists Y (\varphi_1 \wedge \varphi_2 \wedge \varphi_3)$, which can be converted to the form of $\exists Y \forall \bar{x} (\exists \bar{y} Y \bar{z} \bar{y} \wedge C_1 \wedge \cdots \wedge C_m)$, where $\bar{z}$ are variables from $\bar{x}$. □

Proposition 3.3. *Every second-order formula is equivalent to an* SO-KROM^r *formula. More precisely, for each $k \geq 1$, if k is even, then $\Sigma^1_k \leq \Sigma^1_{k+1}$*-KROM^r*; and if k is odd, then $\Pi^1_k \leq \Pi^1_{k+1}$*-KROM^r.

Proof. Given a Σ^1_k-formula $\exists X_1 \forall X_2 \ldots \exists X_{k-1} \forall X_k \varphi$, where k is even and φ does not contain second-order quantifiers, we show that it is equivalent to a Σ^1_{k+1}-KROM^r formula. The proof for the other cases is essentially the same. By Lemma 3.2, $\neg\varphi$ is equivalent to a formula $\exists X_{k+1} \forall \bar{x} (\exists \bar{y} X_{k+1} \bar{z} \bar{y} \wedge C_1 \wedge \cdots \wedge C_m)$. So $\exists X_1 \forall X_2 \ldots \exists X_{k-1} \forall X_k \varphi$ is equivalent to the formula

$$\Phi = \exists X_1 \forall X_2 \ldots \exists X_{k-1} \forall X_k \forall X_{k+1} \exists \bar{x} (\forall \bar{y} \neg X_{k+1} \bar{z} \bar{y} \vee D_1 \vee \cdots \vee D_m)$$

where each D_j is a conjunction of atomic (or negated atomic) formulas. Suppose that Φ is over vocabulary σ. Given a σ-structure $\mathcal{A}$, we construct a Σ_k-DNF quantified Boolean formula ψ such that $\mathcal{A} \models \Phi$ iff ψ is true. Let A be the domain of $\mathcal{A}$. We replace the first-order part $\exists \bar{x} (\forall \bar{y} \neg Y \bar{z} \bar{y} \vee D_1 \vee \cdots \vee D_m)$ by

$$\bigvee_{\bar{a} \in A^{|\bar{x}|}} \Big(\Big(\bigwedge_{\bar{b} \in A^{|\bar{y}|}} \neg Y \bar{z} \bar{y} [\bar{y}/\bar{b}] \Big) \vee D_1 \vee \cdots \vee D_m \Big) [\bar{x}/\bar{a}].$$

We remove the clauses with a formula $(\neg) R \bar{c}$ that is false in $\mathcal{A}$ and delete the formulas $(\neg) R \bar{c}$ that are true in $\mathcal{A}$ in every clause, where $R \in \sigma$. Then we replace each quantifier $\exists X_i$ (or $\forall X_i$) with a sequence $\exists X_i \bar{d}_1 \ldots \exists X_i \bar{d}_{|A|^{arity(X_i)}}$ (or $\forall X_i \bar{d}_1 \ldots \forall X_i \bar{d}_{|A|^{arity(X_i)}}$) where each $\bar{d}_j \in A^{arity(X_i)}$. We treat the atoms $X_i \bar{d}_j$ as propositional variables, and the resulting formula ψ is a Σ^1_k-DNF quantified Boolean formula. It is clear that $\mathcal{A} \models \Phi$ iff ψ is true.

By Proposition 3.1 and its proof, we know that ψ can be encoded in a τ-structure $\mathcal{B}$, where $\tau = \langle \text{Clause}, \text{Var}_1, \ldots, \text{Var}_k, \text{Pos}, \text{Neg} \rangle$ and there is a Σ^1_{k+1}-KROM$^r(\tau)$ formula Ψ such that ψ is true iff $\mathcal{B} \models \Psi$. In the following, we define a quantifier-free interpretation

$$\Pi = \Big(\pi_{\text{uni}}(\bar{v}), \pi_{\text{Clause}}(\bar{v}), \pi_{\text{Var}_1}(\bar{v}), \ldots, \pi_{\text{Var}_k}(\bar{v}), \pi_{\text{Pos}}(\bar{v}_1, \bar{v}_2), \pi_{\text{Neg}}(\bar{v}_1, \bar{v}_2) \Big)$$

of τ in σ, where $\pi_{\text{uni}}, \pi_{\text{Clause}}, \pi_{\text{Var}_1}, \ldots, \pi_{\text{Var}_k}, \pi_{\text{Pos}}, \pi_{\text{Neg}}$ are all quantifier-free formulas over σ. Intuitively, π_{uni} defines the domain of $\mathcal{B}$, π_{Clause} defines the set of clauses of ψ, each π_{Var_i} ($1 \leq i \leq k$) defines the set of variables occurring in the quantifier block $\exists(\forall) X_i$, π_{Pos} (or π_{Neg}) defines a variable occurs positively (or negatively) in a clause.

For any σ-structure $\mathcal{A}$, Π defines a τ-structure $\mathcal{A}^\Pi$ that encodes the formula ψ such that $\mathcal{A}^\Pi \models \Psi$ iff ψ is true iff $\mathcal{A} \models \Phi$. Since Π is an interpretation of τ in σ, we can construct a Σ^1_{k+1}-KROM$^r(\sigma)$ formula $\Psi^{-\Pi}$ from Ψ such that $\mathcal{A}^\Pi \models \Psi$ iff $\mathcal{A} \models \Psi^{-\Pi}$. Therefore, $\Psi^{-\Pi}$ and Φ are equivalent. For more details of the interpretation from one vocabulary to another, we refer the reader to [EF95].

We suppose that $\mathcal{A}$ contains at least two different elements. Let

$$g = \max\{arity(X_1), \ldots, arity(X_k), arity(X_{k+1}))\},$$
$$d = 3 + \max\{(|\bar{x}| + m + 1), (g + k + 1)\}.$$

Define the width of Π to be d. Let $\pi_{\mathrm{uni}}(\bar{v}) = \bigwedge_{i=1}^{d}(v_i = v_i)$, it defines the domain of $\mathcal{A}^\Pi$. For any $\bar{a} = (a_1, a_2, \ldots, a_d) \in A^d$, we will make the following assumptions:

- if $\bar{a}$ encodes a clause, then $a_1 \neq a_3 \wedge a_2 = a_3$, and
- if $\bar{a}$ encodes a variable, then $a_1 \neq a_3 \wedge a_1 = a_2$.

If $\bar{a}$ encodes a clause, it is partitioned as follows

$$\underbrace{a_1a_2a_3}_{a_2=a_3}\underbrace{a_4\cdots a_{m+4}}_{m+1}\underbrace{a_{m+5}\cdots a_{m+4+|\bar{x}|}}_{|\bar{x}|}\underbrace{a_{m+5+|\bar{x}|}\cdots a_d}_{\text{padding elements}}$$

where $a_1 \neq a_3, a_2 = a_3$, and $a_{m+5}, \ldots, a_{m+4+|\bar{x}|}$ are interpretations for $\bar{x}$. $a_4, \ldots, a_{m+4}$ are used to encode the clauses $\forall\bar{y}\neg X_{k+1}\bar{z}\bar{y}$, $D_1, \ldots, D_m$. More precisely, we use $a_1 = a_4 \wedge \bigwedge_{5\leq j\leq m+4} a_3 = a_j$ to indicate that $\bar{a}$ encodes the clause $\forall\bar{y}\neg X_{k+1}\bar{z}\bar{y}$, and use $\bigwedge_{4\leq j\leq i+4} a_1 = a_j \wedge \bigwedge_{i+5\leq h\leq m+4} a_3 = a_h$ to indicate that $\bar{a}$ encodes D_i for $i \in \{1, \ldots, m\}$, respectively. This can be expressed by the formula

$$\Big(v_1 = v_4 \wedge \bigwedge_{5\leq j\leq m+4} v_3 = v_j\Big) \vee \bigvee_{1\leq i\leq m}\Big(\bigwedge_{4\leq j\leq i+4} v_1 = v_j \wedge \bigwedge_{i+5\leq h\leq m+4} v_3 = v_h\Big). \tag{3.1}$$

We also require that there is no formula that is false in clause D_i $(1 \leq i \leq m)$, when $\bar{x}$ are interpreted by $a_{m+5}\cdots a_{m+4+|\bar{x}|}$. Let α_i denote the first-order part of D_i. This is can be expressed by the formula

$$\bigvee_{1\leq i\leq m}\Big(\Big(\bigwedge_{4\leq j\leq i+4} v_1 = v_j \wedge \bigwedge_{i+5\leq h\leq m+4} v_3 = v_h\Big) \to \alpha_i[\bar{x}/v_{m+5}\cdots v_{m+4+|\bar{x}|}]\Big). \tag{3.2}$$

All padding elements must equal a_1, which can be expressed by

$$v_1 \neq v_3 \wedge v_2 = v_3 \wedge \bigwedge_{m+5+|\bar{x}|\leq i\leq d} v_1 = v_i \tag{3.3}$$

Define $\pi_{\mathrm{Clause}}(\bar{v})$ to be the conjunction of (3.1), (3.2) and (3.3).

If $\bar{a}$ encodes a variable, it is partitioned as follows

$$\underbrace{a_1a_2a_3}_{a_1=a_2}\underbrace{a_4\cdots a_{k+4}}_{k+1}\underbrace{a_{k+5}\cdots a_{k+4+arity(X_i)}}_{artiy(X_i)}\underbrace{a_{k+5+arity(X_i)}\cdots a_d}_{\text{padding elements}}$$

where $a_1 \neq a_3, a_1 = a_2$, and $a_4\cdots a_{k+4}$ encode $X_1, \ldots, X_{k+1}$. More precisely, we use $\bigwedge_{4\leq j\leq i+3} a_1 = a_j \wedge \bigwedge_{i+4\leq h\leq k+4} a_3 = a_h$ to indicate that $\bar{a}$ encodes the variable with relation symbol X_i, where $(1 \leq i \leq k+1)$. This can be expressed by the following formula

$$\mathrm{Var}_i(\bar{v}) = \Big(\bigwedge_{4\leq j\leq i+3} v_1 = v_j \wedge \bigwedge_{i+4\leq h\leq k+4} v_3 = v_h\Big).$$

We use $a_{k+5}\cdots a_{k+4+arity(X_i)}$ to indicate that $\bar{a}$ encodes the atom $X_i a_{k+5}\cdots a_{k+4+arity(X_i)}$. We also require that all padding elements $a_{k+5+arity(X_i)}\cdots a_d$ equal a_1. The formula $\pi_{\mathrm{Var}_i}(\bar{v})$, for $i\in\{1,\dots,k-1\}$, is defined by

$$\pi_{\mathrm{Var}_i}(\bar{v})=\Big(v_1\neq v_3\wedge v_1=v_2\wedge \mathrm{Var}_i(\bar{v})\wedge \bigwedge_{k+5+arity(X_i)\le j\le d} v_1=v_j\Big).$$

Define $\pi_{V_k}(\bar{v})$ to be the conjunction of $v_1\neq v_3\wedge v_1=v_2$ and

$$\Big(\mathrm{Var}_k(\bar{v})\wedge \bigwedge_{k+5+arity(X_k)\le j\le d} v_1=v_j\Big)\vee\Big(\mathrm{Var}_{k+1}(\bar{v})\wedge \bigwedge_{k+5+arity(X_{k+1})\le j\le d} v_1=v_j\Big).$$

In the following we define the formula $\pi_{\mathrm{Pos}}(\bar{v}_1,\bar{v}_2)$, which expresses that the atom encoded by $\bar{v}_2$ occurs positively in clause D_j ($1\le j\le m$) encoded by $\bar{v}_1$. Let $\bar{v}_1=v_{1,1}\dots v_{1,d}$ and $\bar{v}_2=v_{2,1}\dots v_{2,d}$. We use the following formula $\varphi_{D_j}(\bar{v}_1)$ to express that $\bar{v}_1$ encodes clause D_j for $j\in\{1,\dots,m\}$.

$$\varphi_{D_j}(\bar{v}_1)=\Big(\pi_{\mathrm{Clause}}(\bar{v}_1)\wedge\big(\bigwedge_{4\le l\le j+4} v_{1,1}=v_{1,l}\wedge\bigwedge_{j+5\le h\le m+4} v_{1,3}=v_{1,h}\big)\Big).$$

Suppose that the atomic formula $X_i\bar{x}'$ occurs in clause D_j, where $\bar{x}'=x'_1\dots x'_{arity(X_i)}$ are variables from $\bar{x}$, and $\bar{v}_2$ encodes the atom $X_i\bar{v}'_2$ where $\bar{v}'_2$ are the corresponding elements in $\bar{v}_2$ by its definition. Let $\bar{v}'_1$ be obtained by replacing $\bar{x}'$ with the corresponding elements in $\bar{v}_1$ that encodes D_j (note that $v_{1,m+5},\dots,v_{1,m+4+|\bar{x}|}$ are interpretations for $\bar{x}$, and $X_i\bar{x}'[\bar{x}/(v_{1,m+5},\dots,v_{1,m+4+|\bar{x}|})]=X_i\bar{v}'_1$). We require that $X_i\bar{v}'_1=X_i\bar{v}'_2$, i.e., $\bar{v}'_1=\bar{v}'_2$. The following formula $\alpha_{D_j,X_i}(\bar{v}_1,\bar{v}_2)$ expresses that the atom encoded by $\bar{v}_2$ occurs positively in clause D_j encoded by $\bar{v}_1$. For $i\in\{1,\dots,k-1\}$, define

$$\alpha_{D_j,X_i}(\bar{v}_1,\bar{v}_2)=\Big(\varphi_{D_j}(\bar{v}_1)\wedge\pi_{\mathrm{Var}_i}(\bar{v}_2)\wedge\bigvee_{X_i\bar{x}'\text{ occurs positively in }D_j}\bar{v}'_1=\bar{v}'_2\Big),$$

and for $i\in\{k,k+1\}$, define

$$\alpha_{D_j,X_i}(\bar{v}_1,\bar{v}_2)=\Big(\varphi_{D_j}(\bar{v}_1)\wedge\pi_{\mathrm{Var}_k}(\bar{v}_2)\wedge\mathrm{Var}_i(\bar{v}_2)\wedge\bigvee_{X_i\bar{x}'\text{ occurs positively in }D_j}\bar{v}'_1=\bar{v}'_2\Big).$$

Define $\pi_{\mathrm{Pos}}(\bar{v}_1,\bar{v}_2)$ to be the conjunction of $v_{1,1}=v_{2,1}\wedge v_{1,3}=v_{2,3}$ and

$$\bigvee\{\alpha_{D_j,X_i}(\bar{v}_1,\bar{v}_2)\mid X_i\text{ has a positive occurrence in }D_j\}.$$

Similarly, we can define the formula $\pi'_{\mathrm{Neg}}(\bar{v}_1,\bar{v}_2)$ to express that the atom encoded by $\bar{v}_2$ occurs negatively in clause D_j ($1\le j\le m$) encoded by $\bar{v}_1$. For the clause $\forall\bar{y}\neg X_{k+1}\bar{z}\bar{y}$, let $\bar{v}''_1$ and $\bar{v}''_2$ be obtained by replacing $\bar{z}$ with the corresponding elements in $\bar{v}_1$ that encodes the clause, and the corresponding elements in $\bar{v}_2$ that encodes X_{k+1}, respectively. Let

$$\beta_{X_{k+1}}(\bar{v}_1,\bar{v}_2)=\left(\begin{array}{l}\pi_{\mathrm{Clause}}(\bar{v}_1)\wedge(v_{1,1}=v_{1,4}\wedge\bigwedge_{5\le i\le m+4}v_{1,3}=v_{1,i})\wedge\\ \pi_{\mathrm{Var}_k}(\bar{v}_2)\wedge\mathrm{Var}_{k+1}(\bar{v}_2)\wedge v_{1,1}=v_{2,1}\wedge v_{1,3}=v_{2,3}\wedge\bar{v}''_1=\bar{v}''_2\end{array}\right).$$

Define $\pi_{\mathrm{Neg}}(\bar{v}_1,\bar{v}_2)=\pi'_{\mathrm{Neg}}(\bar{v}_1,\bar{v}_2)\vee\beta_{X_{k+1}}(\bar{v}_1,\bar{v}_2)$.

Let Θ be the following formula

$$\exists Z_1 \forall Z_2 \ldots \exists Z_{k-1} \forall Z_k \exists Y \forall \bar{v}_1 \forall \bar{v}_2 \left(\begin{array}{l} \exists \bar{z} Y(\bar{z}) \wedge \Big(Y(\bar{v}_1) \rightarrow \pi_{\text{Clause}}(\bar{v}_1) \Big) \wedge \\ \bigwedge_{1 \leq i \leq k} \Big(Y(\bar{v}_1) \wedge \pi_{\text{Pos}}(\bar{v}_1, \bar{v}_2) \wedge \pi_{\text{Var}_i}(\bar{v}_2) \rightarrow Z_i(\bar{v}_2) \Big) \wedge \\ \bigwedge_{1 \leq i \leq k} \Big(Y(\bar{v}_1) \wedge \pi_{\text{Neg}}(\bar{v}_1, \bar{v}_2) \wedge \pi_{\text{Var}_i}(\bar{v}_2) \rightarrow \neg Z_i(\bar{v}_2) \Big) \end{array} \right).$$

The formula Θ says that there is a valuation for X_1, for any valuation to X_2, ..., there is a valuation to X_{k-1}, for any valuation to X_k and X_{k+1}, there is a nonempty set Y of clauses, such that all literals in the clauses are true under the valuation. Φ and Θ are equivalent on the structures with at least two elements. For any finite structure, there is a quantifier-free formula that captures its isomorphism type [EF95]. So on one-element structures, Φ is equivalent to $\forall x \forall y (x = y \wedge \delta(x))$, where $\delta(x)$ is a disjunction of isomorphism types of one-element structures satisfying Φ. The formulas $\Theta \vee \forall x \forall y (x = y \wedge \delta(x))$ and Φ are equivalent on all finite structures. Since all formulas in Π are quantifier-free, $\Theta \vee \forall x \forall y (x = y \wedge \delta(x))$ can be converted to an equivalent Σ^1_{k+1}-KROM^r formula by elementary techniques. □

The following proposition says that the data complexity of SO-KROM^r is in the polynomial hierarchy.

Proposition 3.4. *For each $k \geq 1$, if k is odd, then the data complexity of Π^1_{k+1}-KROM^r and Π^1_{k+2}-KROM^r are in Π^p_k; if k is even, then the data complexity of Σ^1_{k+1}-KROM^r and Σ^1_{k+2}-KROM^r are in Σ^p_k.*

Proof. From Corollary 2.2, we know that Π^1_{k+1}-KROM$^r \equiv \Pi^1_{k+2}$-KROM^r if k is odd, and Σ^1_{k+1}-KROM$^r \equiv \Sigma^1_{k+2}$-KROM^r if k is even. We only prove that the data complexity of Π^1_{k+1}-KROM^r is in Π^p_k (k is odd), the proof for the other cases is similar.

Let $\Phi = \forall \overline{X}_1 \exists \overline{X}_2 \ldots \forall \overline{X}_k \exists \overline{X}_{k+1} \forall \bar{x} \varphi$ be a Π^1_{k+1}-KROM^r formula (k is odd). Given a structure $\mathcal{A}$, we construct an alternating Turing machine that first assigns the values of $\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_k$ alternately between universal and existential moves according to their quantifier types. This step can be done in Π^p_k. The complexity of deciding whether $(\mathcal{A}, \overline{X}^A_1, \overline{X}^A_2, \ldots, \overline{X}^A_k) \models \exists \overline{X}_{k+1} \forall \bar{x} \varphi$ is in NL, since all occurrences of $\exists \bar{z} X_i \bar{z}$ ($1 \leq i \leq k$) in $\exists \overline{X}_{k+1} \forall \bar{x} \varphi$ can be replaced by their truth values, and the resulting formula is a Σ^1_1-KROM^r formula which can be evaluated in NL by Proposition 2.5. Therefore, the total complexity of checking $\mathcal{A} \models \Phi$ is in Π^p_k. □

Since Σ^1_k captures Σ^p_k and Π^1_k captures Π^p_k for $k \geq 1$, combining Proposition 3.3 with Proposition 3.4 we conclude the following corollary.

Corollary 3.5. *On all finite structures, for each $k \geq 1$, if k is even, then Σ^1_{k+1}-KROM$^r \equiv \Sigma^1_k$, and if k is odd, then Π^1_{k+1}-KROM$^r \equiv \Pi^1_k$.*

Theorem 3.6. *On all finite structures, for each $k \geq 1$, if k is even, then Σ^1_{k+1}-KROM^r captures Σ^p_k, and if k is odd, then Π^1_{k+1}-KROM^r captures Π^p_k.*

4. An extended version of second-order Krom logic

In this section, we define second-order extended Krom logic and study its expressive power and data complexity.

Definition 4.1. Second-order extended Krom logic over a vocabulary τ, denoted by SO-EKROM(τ), is the set of second-order formulas of the form

$$\forall X_1 \exists Y_1 \cdots \forall X_k \exists Y_k \forall \bar{x}(C_1 \wedge \cdots \wedge C_n),$$

where C_i $(1 \leq i \leq n)$ are extended Krom clauses with respect to $Y_1, \ldots, Y_k$, more precisely, each C_i is a disjunction of the form

$$\alpha_1 \vee \cdots \vee \alpha_l \vee H_1 \vee H_2,$$

where

(1) each α_s is either $Q\bar{y}$ or $\neg Q\bar{y}$, where $Q \in \tau \cup \{X_1, \ldots, X_k\}$,
(2) each H_t is either $Y_i\bar{z}$ or its negation $\neg Y_i\bar{z}$, where $(1 \leq i \leq k)$.

Proposition 4.2. SO-EKROM *is closed under substructures.*

Proof. All universal first-order formulas are closed under substructures. It is easy to check that the formula obtained by quantifying a relation in a formula which is closed under substructures still preserves this property. □

Proposition 4.3. *The data complexity of* SO-EKROM *is in* co-NP*.*

Proof. Let $\Phi = \forall X_1 \exists Y_1 \ldots \forall X_k \exists Y_k \forall \bar{x}(C_1 \wedge \cdots \wedge C_n)$ be a SO-EKROM formula over σ. For an arbitrary σ-structure $\mathcal{A}$, we replace the first-order part $\forall \bar{x}(C_1 \wedge \cdots \wedge C_n)$ by $\bigwedge_{\bar{a} \in A^{|\bar{x}|}}(C_1 \wedge \cdots \wedge C_n)[\bar{x}/\bar{a}]$. We remove the clauses with a formula $(\neg)R\bar{b}$ that is true in $\mathcal{A}$ and delete the formulas $(\neg)R\bar{b}$ that are false in $\mathcal{A}$ in every clause, where R is a relation symbol in σ. Then we replace each second-order quantifier $\forall X_i$ $(\exists Y_i)$ in the prefix with a sequence $\forall X_i\bar{d}_1 \ldots \forall X_i\bar{d}_{|A|^{arity(X_i)}}$ $(\exists Y_i\bar{d}_1 \ldots \exists Y_i\bar{d}_{|A|^{arity(X_i)}})$ where each $\bar{d}_j \in A^{arity(X_i)}$. We treat the atoms $X_i\bar{d}_j$ $(Y_i\bar{d}_j)$ as propositional variables, the resulting formula ψ is a QE-2CNF formula. It is clear that $\mathcal{A} \models \Phi$ iff ψ is true. It was proved that for any fixed number m, the evaluation problem for the QE-2CNF formulas whose quantifier prefixes have m alternations is in co-NP [FKB90]. Hence, whether $\mathcal{A} \models \Phi$ holds is decidable in co-NP. □

Since Π^1_1 captures co-NP, we can get the following corollary.

Corollary 4.4. *Every* SO-EKROM *formula is equivalent to a* Π^1_1 *formula on ordered finite structures.*

Proposition 4.5. Π^1_2-KROM$^r \leq \Pi^1_2$-EKROM *on ordered finite structures.*

Proof. Let $\forall\overline{X}\exists\overline{Y}\forall\bar{x}\varphi$ be a Π^1_2-KROM^r formula. We see that $\exists\overline{Y}\forall\bar{x}\varphi$ is a Σ^1_1-KROM^r formula. By Corollary 2.6, it is equivalent to a Σ^1_1-KROM formula on ordered finite structures. This implies that $\forall\overline{X}\exists\overline{Y}\forall\bar{x}\varphi$ is equivalent to a Π^1_2-EKROM formula on ordered finite structures. □

Combining Proposition 3.3, Corollary 4.4 and Proposition 4.5 gives the following corollary.

Corollary 4.6. SO-EKROM $\equiv \Pi^1_2$-EKROM $\equiv \Pi^1_1$ *on ordered finite structures.*

Theorem 4.7. *Both* SO-EKROM *and* Π^1_2-EKROM *can capture* co-NP *on ordered finite structures.*

5. Conclusion

In this paper, we introduce second-order revised Krom logic and study its expressive power and data complexity. SO-KROM^r is an extension of SO-KROM by allowing $\exists \bar{z} R \bar{z}$ in the formula matrix, where R is a second-order variable. For SO-KROM^r, we show that the innermost universal second-order quantifiers can be removed. Hence, Σ^1_k-KROM$^r \equiv \Sigma^1_{k+1}$-KROM^r for odd k, and Π^1_k-KROM$^r \equiv \Pi^1_{k+1}$-KROM^r for even k. SO-KROM collapses to its existential fragment. The same statement is unlikely to be true for SO-KROM^r. On ordered finite structures, we prove that Σ^1_1-KROM^r equals Σ^1_1-KROM, and captures NL. On all finite structures, we show that $\Sigma^1_k \equiv \Sigma^1_{k+1}$-KROM^r for even k, and $\Pi^1_k \equiv \Pi^1_{k+1}$-KROM^r for odd k. This result gives an alternative logic for capturing the polynomial hierarchy, which is the main contribution of the paper. We also study an extended version of second-order Krom logic SO-EKROM. On ordered finite structures, SO-EKROM collapses to Π^1_2-EKROM and equals Π^1_1. Therefore, both of them can capture co-NP on ordered finite structures.

References

[ABD09] Albert Atserias, Andrei Bulatov, and Anuj Dawar. Affine systems of equations and counting infinitary logic. *Theoretical Computer Science*, 410(18):1666–1683, 2009.

[AV89] Serge Abiteboul and Victor Vianu. Fixpoint extensions of first-order logic and Datalog-like languages. In *Proceedings. Fourth Annual Symposium on Logic in Computer Science*, pages 71–72. IEEE Computer Society, 1989.

[CK04] S. Cook and A. Kolokolova. A second-order theory for NL. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science, 2004.*, pages 398–407, 2004. `doi:10.1109/LICS.2004.1319634`.

[Daw08] Anuj Dawar. On the descriptive complexity of linear algebra. In *International Workshop on Logic, Language, Information, and Computation*, pages 17–25. Springer, 2008.

[DGHL09] Anuj Dawar, Martin Grohe, Bjarki Holm, and Bastian Laubner. Logics with rank operators. In *Logic In Computer Science, 2009. LICS'09. 24th Annual IEEE Symposium on*, pages 113–122. IEEE, 2009.

[DGP19] Anuj Dawar, Erich Grädel, and Wied Pakusa. Approximations of isomorphism and logics with linear-algebraic operators. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 112:1–112:14, 2019.

[EF95] Heinz-Dieter Ebbinghaus and Jörg Flum. *Finite model theory.* Springer-Verlag Berlin Heidelberg, 1995.

[Fag74] R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. *SIAM-AMS Proceedings*, 7:43–73, 1974.

[FKB90] Andreas Flögel, Marek Karpinski, and Hans Kleine Büning. Subclasses of quantified boolean formulas. In *Proceedings of the 4th Workshop on Computer Science Logic*, CSL 1990, page 145–155, Berlin, Heidelberg, 1990. Springer-Verlag.

[FZ12] Shiguang Feng and Xishun Zhao. The complexity and expressive power of second-order extended logic. *Studies in Logic*, 5(1):11–34, 2012.

[FZ13] Shiguang Feng and Xishun Zhao. Complexity and expressive power of second-order extended Horn logic. *Mathematical Logic Quarterly*, 59(1-2):4–11, 2013.

[GO92] Erich Grädel and Martin Otto. Inductive definability with counting on finite structures. In *Selected Papers from the Workshop on Computer Science Logic*, CSL '92, page 231–247, Berlin, Heidelberg, 1992. Springer-Verlag.

[Grä91] Erich Grädel. The expressive power of second order Horn logic. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 466–477. Springer, 1991.

[Grä92] Erich Grädel. Capturing complexity classes by fragments of second-order logic. *Theoretical Computer Science*, 101(1):35–57, 1992.

[Imm82] Neil Immerman. Relational queries computable in polynomial time. In *fourteenth annual ACM symposium on Theory of computing*, pages 147–152. ACM, 1982.

[Imm87] Neil Immerman. Languages that capture complexity classes. *SIAM Journal on Computing*, 16(4):760–778, 1987.

[Imm98] Neil Immerman. *Descriptive Complexity*. Springer Verlag, 1998.

[Ott96] Martin Otto. The expressive power of fixed-point logic with counting. *Journal of Symbolic Logic*, pages 147–176, 1996.

[Sto76] Larry J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.

[Var82] Moshe Y Vardi. The complexity of relational query languages. In *fourteenth annual ACM symposium on Theory of computing*, pages 137–146. ACM, 1982.