7. Monadic Second-Order Logic and Interpretability

For modelling it is convenient to assume
the (undirected) graphs to be vertex and edge labelled:

$$G = (V, E, \lambda_V, \lambda_E)$$

with E = set of pairs of vertices $\{x, y\}$
$\lambda_V : V \to \Sigma_V$ vertex labelling
$\lambda_E : E \to \Sigma_E$ edge labelling.

7.1 MSO

Goal: Introduce a logical language
to talk about properties of such graphs

Recall (from logic):
↳ A signature is a pair $Sig = (Fun, Pred)$
of function symbols f/n in the set Fun and
predicate symbols p/n in the set $Pred$.
Here, n denotes the arity of a symbol,
the number of arguments that are expected.
↳ To talk about graphs, need purely relational signatures,
which means $Fun = \emptyset$.
We refer to such signatures as
$$Sig = \{p_1/n_1, p_2/n_2, \dots\}$$

Definition ($MSO[Sig]$)
Consider two countably infinite sets
$V_1 = \{x, y, \dots\}$ of first-order variables and
$V_2 = \{X, Y, \dots\}$ of second-order variables
↳ MSO

The set of formulas in MSO over Sig, denoted MSO[Sig], is defined by

$$\varphi ::= x_1 = x_2 \mid P(\bar{x}) \mid X(x) \mid \varphi_1 \vee \varphi_2 \mid \neg\varphi$$
$$\mid \exists x.\varphi \mid \exists X.\varphi,$$

where $P \in Sig$ is a predicate symbol of arity $n = |\bar{x}|$, $\bar{x} = x_1, \dots, x_n$.

Intuitively:
↳ First-order variables = elements from domain.
↳ Second-order variables = sets of elements from domain.
↳ $X(x)$ = element x is in set X.

Bound and free variables:
↳ Quantifiers $\exists x$ and $\exists X$ bind variables x and X, respectively.
↳ A variable that is not in the scope of a quantifier is free.

Definition (Sig-structure):
- A Sig-structure is a tuple

$$S = (Dom^S, (P^S)_{P \in Sig})$$

with

Dom^S = the domain of elements (to talk about and quantify over)

$P^S \subseteq \underbrace{Dom^S \times \dots \times Dom^S}_{\text{arity}(P)\text{-times}}$ = the interpretation of predicate symbol P, which is a relation on Dom^S.

- We sometimes consider restricted classes of Sig-structures, for example rather than all graphs only graphs corresponding to pushdown computations.
↳ Refer to such classes as K.

Graphs:

In the case of graphs $G = (V, E, \lambda_V, \lambda_E)$,
we have

$$Sig_G = \{P_V/1, P_E/1, (P_a/1)_{a \in \Sigma_V}, (Q_a/1)_{a \in \Sigma_E}, R/2\}.$$

Moreover, G gives rise to the Sig_G-structure

$$S_G := (Dom^G, P_V^G, P_E^G, (P_a^G)_{a \in \Sigma_V}, (Q_a^G)_{a \in \Sigma_E}, R^G)$$

with

$Dom^G := V \uplus E$

$x \in P_V^G$ iff $x \in V$ // is a vertex

$e \in P_E^G$ iff $e \in E$ // is an edge

$x \in P_a^G$ iff $\lambda_V(x) = a$ // x is labelled a (and is a vertex) for all $a \in \Sigma_V$

$e \in Q_a^G$ iff $\lambda_E(e) = a$ // e is labelled a (and is an edge) for all $a \in \Sigma_E$

$(x, e) \in R^G$ iff $x \in e = \{x, y\}$ // x and e are incident.

__Definition (Satisfaction relation $\models$ for MSO):__

Let S be a Sig-structure and $\varphi \in MSO[Sig]$.
Let $I : V_1 \uplus V_2 \to Dom^S \cup \mathbb{P}(Dom^S)$ be an interpretation of the free variables.
Then satisfaction $S, I \models \varphi$ is defined as follows

$S, I \models x_1 = x_2$, if $I(x_1) = I(x_2)$

$S, I \models P(\bar{x})$, if $I(\bar{x}) = (I(x_1), \dots, I(x_n)) \in P^S$

$S, I \models \varphi_1 \vee \varphi_2$, if $S, I \models \varphi_1$ or $S, I \models \varphi_2$

$S, I \models \neg\varphi$, if not $S, I \models \varphi$

$S, I \models \exists x : \varphi$, if <u>there is</u> $m \in Dom^S$ so that $S, I[m/x] \models \varphi$

$S, I \models \exists X : \varphi$, if there is $M \subseteq Dom^S$ so that $S, I[M/X] \models \varphi$.

Here, $(I[m/x])(y) := I(y)$ if $y \neq x$ and $(I[m/x])(x) := m$. Similar for X.

Interested in <u>closed formulas</u> (without free variables, also called <u>sentences</u>)

↳ Meaning does not depend on interpretation I of variables

↳ Still need I for truth of subformulas.

- Closed formula φ is <u>satisfiable</u> (in class K)

  if $\quad S \models \varphi \quad$ for some Sig-structure (from K)

  Call S a <u>model</u> of φ.
- Formula without model is unsatisfiable.
- Formula with $S \models \varphi$ for all Sig-structures (from K)

  is <u>valid</u> (in K)

<u>Note:</u>

φ valid iff $\neg\varphi$ unsatisfiable.

<u>Language</u> defined by φ is

$$L(\varphi) := \{ S \text{ Sig-structure (from } K) \mid S \models \varphi \}.$$

<u>Definition:</u>

A property P is a <u>monadic second-order</u> property

(over a class K) of Sig-structures,

if there is $\varphi \in MSO[Sig]$

so that for all Sig-structures S $(\in K)$:

P holds for S iff $S \models \varphi$.

<u>Examples:</u>

(1) <u>k-colorability</u>: $\exists X_1, \dots, X_k : \forall y : P_V(y) \rightarrow \bigvee_{i=1}^{k} X_i(y)$

$$\wedge \bigwedge_{i \neq j} \neg \exists x : P_V(x) \wedge X_i(x) \wedge X_j(x)$$

$$\wedge \forall x, y : \neg(x = y) \wedge edge(x,y)$$

$$\rightarrow \neg \bigvee_{i=1}^{k} X_i(x) \wedge X_i(y)$$

Here $edge(x,y) := \exists e : R(x,e) \wedge R(y,e)$.

(2) Reachability:

$$reach(x,y) := \forall X [X(x) \wedge (\forall z_1, z_2 . \, edge(z_1,z_2) \wedge X(z_1) \to X(z_2)) \to X(y)].$$

7.2 Interpretability

Goal:
- Transform algorithmic problem P_1 on a class of structures K_1 into a different problem P_2 on K_2.
- Typically used in logic to deduce decidability and undecidability results for new theories from better understood theories.

Technically:

Given classes of structures with their MSO-languages
K_1 with $MSO[Sig_1]$ and K_2 with $MSO[Sig_2]$.

- Transform every structure $S \in K_1$ into a structure $A(S) \in K_2$ so that

$$S =_{iso} \tau(A(S)).$$

"If you apply the interpretation τ, then $A(S)$ looks like S."

- Moreover transform every formula $\varphi \in MSO[Sig_1]$ into a new formula $\tau^{-1}(\varphi) \in MSO[Sig_2]$.

Formula $\tau^{-1}(\varphi)$ explains how to evaluate φ on $A(S)$. To this end, it encodes the effect of transformation $A(-)$, which is described by interpretation τ.

Together:

$$S \models \varphi$$

If you understand $A(S)$ under τ, it behaves like S.

$$(S =_{iso} \tau(A(S))) \iff \tau(A(S)) \models \varphi$$

$$(\text{To be shown}) \iff A(S) \models \tau^{-1}(\varphi)$$

Rather than transforming $A(S)$ via τ, let us modify φ so that it takes $\tau(A)$ into account.

Graphically:

$$\begin{array}{ccc} K_1 & \underset{\tau}{\overset{A}{\rightleftarrows}} & K_2 \\ \Pi & & \Pi \\ MSO[Sig_1] & \xrightarrow{\tau^{-1}} & MSO[Sig_2] \end{array}$$

Auxiliary definitions:

Consider a Sig-structure

$$S = (Dom^S, (P^S)_{P \in Sig}).$$

- Moreover, let $\equiv$ be an equivalence on Dom^S.

Then

$$S/_{\equiv} := (Dom^S/_{\equiv}, (P^S/_{\equiv})_{P \in Sig})$$

with

$$([a_1]_{\equiv}, \dots, [a_n]_{\equiv}) \in P^S/_{\equiv}, \text{ if}$$

$$\exists\, a_1' \equiv a_1, \dots, a_n' \equiv a_n : (a_1, \dots, a_n) \in P^S.$$

- Moreover, given $\gamma(x_1, \dots, x_n) \in MSO[Sig]$, we define

$$S(\gamma) := \{ (a_1, \dots, a_n) \in {Dom^S}^n \mid S, I[a_1, \dots, a_n / x_1, \dots, x_n] \models \gamma \}$$

Definition (MSO-Interpretation):

The MSO-interpretation of K_1 over Sig_1 in K_2 over Sig_2 is a tuple of formulas

$$\tau = (\alpha(x), \varepsilon(x_1, x_2), (\gamma_P(x_1, \dots, x_{arity(P)}))_{P \in Sig_1}) \in MSO[Sig_2]^{2 + |Sig_1|}.$$

For every $S \in K_2$, we require $S(\mathcal{E})$ to be an equivalence, and define

$$\tau(S) := \left(S(\alpha), (S(\gamma_P))_{P \in Sig_1}\right) /_{S(\mathcal{E})}.$$

Note:

- $\tau(S)$ is a Sig_1-structure, but need not belong to K_1.
- $\alpha(x)$ explains which elements from $Dom^{A(S)}$ play the role of elements in Dom^S
- $\mathcal{E}(x_1, x_2)$ explains which elements from $Dom^{A(S)}$ to identify
- $\gamma_P(x_1, ..., x_n)$ explains how to evaluate $P(x_1, ..., x_n)$ on $A(S)$

Definition (Interpretability):

K_1 over Sig_1 is (polynomial-time) interpretable in K_2 over Sig_2,

if there is an MSO-interpretation τ of K_1 in K_2

and an algorithm A

that constructs for each $S \in K_1$ a structure $A(S) \in K_2$

(in time polynomial in $|S|$)

so that

$$S =_{MSO} \tau(A(S)).$$

It remains to encode the formulas from $MSO[Sig_1]$ into $MSO[Sig_2]$.

Definition (τ^{-1}):

$$\tau^{-1}(x_1 = x_2) := \mathcal{E}(x_1, x_2)$$

$$\tau^{-1}(X(x)) := \exists y.\ \mathcal{E}(x, y) \land X(y)$$

$$\tau^{-1}(P(x_1, ..., x_n)) := \exists y_1, ..., y_n. \left(\bigwedge_{i=1}^{n} \mathcal{E}(x_i, y_i) \land \gamma_P(y_1, ..., y_n)\right)$$

$$\tau^{-1}(\neg \varphi) := \neg \tau^{-1}(\varphi)$$

$$\tau^{-1}(\varphi_1 \land \varphi_2) := \tau^{-1}(\varphi_1) \land \tau^{-1}(\varphi_2)$$

$$\tau^{-1}(\exists x\, \varphi) := \exists x. (\alpha(x) \wedge \tau^{-1}(\varphi))$$

$$\tau^{-1}(\exists X. \varphi) := \exists X. \tau^{-1}(\varphi).$$

Lemma:

$S \models \varphi$ iff $A(S) \models \tau^{-1}(\varphi)$.

Theorem:

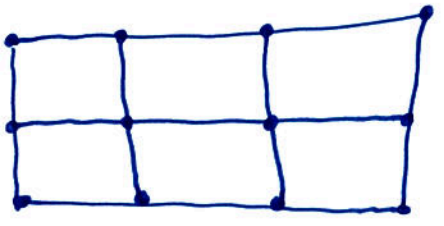
Assume P is an $MSO[Sig_1]$-property on class K_1 and let K_1 be polynomial-time MSO-interpretable in K_2 with the MSO-interpretation τ.
If $\tau^{-1}(P)$ can be solved in K_2 in polynomial time, then P can be solved on K_1 in polynomial time.

Example:

Let K_1 be the class of $3 \times n$ grids.
Let K_2 be the class of finite trees with $\underbrace{3}_{\text{unary}}$ predicates P_1, P_2, P_3
In both structures, the only binary relation is edge(x,y), irreflexive and symmetric.
For the grids, this is the only predicate.

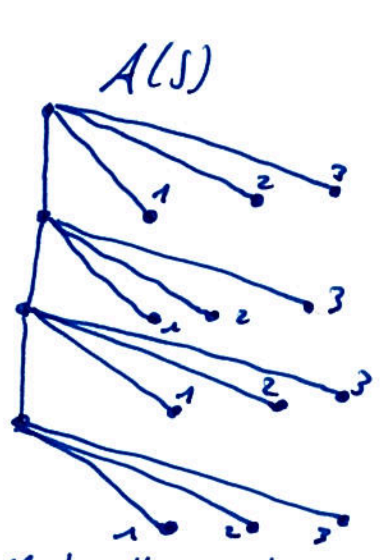
Transformation $A(-)$ turns

S



into

A(S)



Label i indicates that the vertex is in $P_i^{A(S)}$, $1 \leq i \leq 3$

The <u>MSO-interpretation</u> τ_{12}

$$\alpha(x) := \exists y.\ (\neg x = y \wedge edge(x,y) \wedge \neg \exists z.\ \neg z = y \wedge \neg z = x \wedge edge(x,z))$$

$$\varepsilon(x_1, x_2) := x_1 = x_2$$

$$\begin{aligned}\gamma_{edge}(x,y) := \ & \exists z.\ (edge(x,z) \wedge edge(y,z) \\ & \wedge [(P_1(x) \wedge P_2(y)) \vee (P_2(x) \wedge P_1(y)) \\ & \vee (P_2(x) \wedge P_3(y)) \vee (P_3(x) \wedge P_2(y))] \\ & \vee \exists z_1, z_2.\ (edge(x,z_1) \wedge edge(y,z_2) \wedge edge(z_1,z_2) \\ & \wedge [(P_1(x) \wedge P_1(y)) \vee (P_2(x) \wedge P_2(y)) \\ & \vee (P_3(x) \wedge P_3(y))].\end{aligned}$$

Then
$$S =_{iso} \tau(A(S)).$$

Moreover, for every $3 \times n$ grid S in K_1 there is such a tree $A(S)$ in K_2.