PSPACE: a class K of ordered τ-structures is in PSPACE (polynomial space) iœ it is accepted by a deterministic machine M that is p space-bounded for some polynomial $p \in \mathbf{N}[x]$.

B ib lio g r a p h y

[EF95] Heinz-Dieter Ebbinghaus and Jrg Flum. Finite Model Theory. Springer, 1995.

[Eve79] Shimon Even. Graph Algorithms. Computer Science Press, 1979.

[Gav72] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. SIAM Journal of Computing, 1:180˘187, 1972.

[GJ79] Michael R. Garey and David S. Johnson. Computers and Intractability - A Guide to the Theory of NP-Completeness. W.H.Freeman and Co., 1979.

[Pap94] Christos H. Papadimitriou. Computational Complexity. Addison-Wesley, 1994.

K-colorability: Let $G(V, E)$ be a graph, and positive integer $K \leq |V|$. G is K-colorable iœ there exist a function $f : V \to \{1, 2, \ldots, K\}$ such that $f(u) \neq f(v)$ whenever $\{u, v\} \in E$.

vertex cover: Let $G(V, E)$ be a graph, and positive integer $K \leq |V|$. There is a vertex cover of size K or less for G, i.e., a subset $V' \subseteq V$ with $|V'| \leq K$ such that for each edge $\{u, v\} \in E$ at least one of u and v belongs to V'.

independent set: independent set S is a set of vertices such that there is no edge between any pair of vertices from S

bipartite graph: bipartite graph is a graph such that the set of its vertices V can be partitioned into two disjoint sets V_1 and V_2 such that any edge $e \in E$ connects $v' \in V_1$ and $v'' \in V_2$

B Some Complexity Notions

LOGSPACE: a class K of ordered τ-structures is in LOGSPACE (deterministic logarithmic space) iœ there is a deterministic machine M and $d \geq 1$ such that M accepts K and is $d*\log$ space-bounded ($\log n$ stands for the least natural number $\geq \log_2 n$)

NLOGSPACE: a class K of ordered τ-structures is in NLOGSPACE (nondetermimistic logarithmic space) iœ there is a nondeterministic machine M and $d \geq 1$ such that M accepts K and is $d * \log$ space-bounded ($\log n$ stands for the least natural number $\geq \log_2 n$)

PTIME: a class K of ordered τ-structures is in PTIME (polynomial time) iœ there is a deterministic machine M and a polynomial $p \in \mathbf{N}[x]$ such that M accepts K and is p time-bounded.

NPTIME: a class K of ordered τ-structures is in NPTIME (nondeterministic polynomial time) iœ it is accepted by a nondeterministic machine M that is p time-bounded for some polynomial $p \in \mathbf{N}[x]$.

Lemma 2
For chordal graphs the maximum clique size is equal to the minimum coloration size.

We already have IFP deønition for the class of chordal graphs and it is left to state ımaximum clique size is Kȷ. But to describe the class of graphs with maximal cliques of size K, it is enough to say ıthere exist a clique of size K and there is no clique of size $K+1$ȷ. And these two statements are easily expressed in FOL.

$$\varphi_{\text{cl-K}} = \exists v_1, v_2, \ldots, v_k \Big(\bigwedge_{1 \le i < j \le k} \neg v_i = v_j \wedge \bigwedge_{1 \le i < j \le k} E v_i v_j \Big) \tag{16}$$

$$\begin{aligned} \varphi_{\text{no-cl-K+1}} =& \neg \exists v_1, v_2, \ldots, v_k, v_{k+1} \Big(\bigwedge_{1 \le i < j \le k+1} \neg v_i = v_j \\ & \wedge \bigwedge_{1 \le i < j \le k+1} E v_i v_j \Big) \end{aligned} \tag{17}$$

And the ønal sentence that expresses the class of chordal graphs that can be colored using minimum K colors is:

$$\varphi_{\text{chor-K-col}} = \varphi_{\text{chor}} \wedge \varphi_{\text{cl-K}} \wedge \varphi_{\text{no-cl-K+1}} \tag{18}$$

A Graph Theory Notions Used in the Text

Here we bring together all graph notions used in this work.

vertex degree:	vertex degree d of the vertex v for an undirected graph G, is the number of edges incident to v
connectivity:	graph G is connected iœ any pair of its vertices is connected by a path
circuit-free graph:	graph G is circuit-free iœ there is no path that starts and ends in the same vertex of the graph
tree:	graph G is called a tree if it is connected and circuit-free

$v_1 \in C_1$, since C_1 contains vertices that can be added to S, when $S = \emptyset$, that is on the first stage of Algorithm 1 (when no vertices have been eliminated yet). $v_2 \in C_1 \cup C_2$ since either it can be eliminated by Algorithm 1 on the first stage (thus, any stage later), in this case $v_2 \in C_1$, or on the second stage, only after eliminating the first vertex, in this case $v_2 \in C_2$. In general, $v_i \in \bigcup_{1 \leq j \leq i} C_j$ since in the worst case v_i can only be eliminated by Algorithm 1 after $v_1 \in C_1, v_2 \in C_2, \ldots, v_{i-1} \in C_{i-1}$ have been eliminated, in this case $v_i \in C_i$.

The direction (ii): In the analogous way it can be shown that any vertex $v \in C_i$ added to S on the i^{th} iteration of the Algorithm 2 can be eliminated on the i^{th} stage of Algorithm 1 as v_i (or later) by Algorithm 1.

Now we present an IFP formula that captures Algorithm 2.

(14)

$$
\begin{aligned}
\varphi_{\mathrm{ch}}(x) = &[\mathrm{IFP}_{x,X} \\
&(\forall v_1 v_2((\neg v_1 = v_2 \wedge \neg X v_1 \wedge \neg X v_2 \wedge E x v_1 \wedge E x v_2) \Rightarrow E v_1 v_2) \vee \\
&\exists v(\neg X v \wedge E x v \wedge \forall z(\neg z = v \wedge \neg X z \Rightarrow \neg E x z)) \vee \\
&\neg \exists v(\neg X v \wedge E x v))]x
\end{aligned}
$$

First line of the above formula describes vertex x such that all the vertices adjacent to it form a clique. The second line treats special case when x is connected to the only vertex which also forms a clique in empty way.

The final sentence that tests chordality is: $\varphi_{\mathrm{chor}} = \forall x \varphi_{\mathrm{ch}}(x)$

The time complexity of the algorithm for checking chordality that our formula directly captures is:

$$\frac{n(n+1)}{2} \times \frac{d(d-1)}{2} \tag{15}$$

where n is the number of vertices in a graph and d is the maximum degree of a vertex.

3.6 Chordal Graph K-Colorability

We already defined Graph K-Colorability earlier this section. There we treated one instance of this NP-complete problem that belongs to PTIME complexity class, namely the case $K = 2$. Here we present one more PTIME instance based on the following Lemma:

Step 4. If only one vertex of the graph remains uneliminated then the graph is chordal, otherwise not.

We adopt this algorithm to an IFP formula as follows:

Algorithm 2 (for chordality checking)

Step 1. $S \leftarrow \emptyset$.

Step 2. Search for a group of vertices C such that $C \cap S = \emptyset$ and for each vertex v, $v \in C$ it is true that all the vertices u, $u \notin S$ adjacent to it are connected to each other (or form a clique).

Step 3. $S \leftarrow S \cup C$.

Step 4. Repeat Steps 2˘3 until no group of such vertices can be found.

Step 5. If S contains all vertices of the graph then the graph is chordal, otherwise not.

Before we present the IFP formula that directly expresses Algorithm 2 we have to prove the equivalence of Algorithm 1 and Algorithm 2. The general idea of the proof is

(i) to show that every vertex v that Algorithm 1 eliminates will be added to S by Algorithm 2.

(ii) the opposite, every vertex v that the Algorithm 2 adds to S will be eliminated by Algorithm 1.

First, note that Step 4 of Algorithm 1 can be reformulated as: ıIf all vertices of the graph are eliminated then the graph is chordal, otherwise notȷ, since the condition in Step 1 of Algorithm 1 is satisøed by an isolated vertex in empty way.

Second, when Algorithm 2 adds a vertex v to S on some iteration, v with all edges incident to it is not accessible to all the following iterations, that is v is eliminated from the graph with all edges incident to it.

Suppose Algorithm 1 has eliminated vertices in the following order: $v_1, v_2, \ldots, v_{k-1}, v_k$ and Algorithm 2 eliminated groups of vertices in the following order $C_1, C_2, \ldots, C_l$.

The direction (i): Let us show that any v_i eliminated by Algorithm 1 will be added to S by Algorithm 2.

It is easy to see that if V' is both a vertex cover and an independent set for G then $V - V'$ is also both a vertex cover and an independent set for G, that is G is bipartite with one side of size K.

Thus, the class of graphs we want to describe is equivalent to the class of bipartite graphs with one side of size K.

A FOL formula that describes this class of graphs for any given K is:

$$\begin{aligned} &\exists v_1 v_2 \dots v_k [\bigwedge_{1 \leq i < j \leq k} \neg v_i = v_j \wedge \bigwedge_{1 \leq i < j \leq k} \neg E v_i v_j \wedge \\ &\forall xy ((\neg x = y \wedge \bigwedge_{1 \leq i \leq k} \neg v_i = x \wedge \bigwedge_{1 \leq i \leq k} \neg v_i = y) \Rightarrow \neg Exy)] \end{aligned} \tag{13}$$

The ørst line of the above formula describes one side of a bipartite graph of size K saying that there exist K diœerent vertices such that there is no edge between any pair of them. The second line says that for any pair of vertices that does not belong to the already described side of a bipartite there is no edge that connects the pair, thus the second line describes the second side of a bipartite (of size $V - K$). The third line demands existence at least one edge in a graph in order to be a bipartite.

3.5 Chordal Graph

Definition 6
A graph is called chordal if every simple circuit $v_1 - v_2 - v_3 - \dots - v_l - v_1$, with $l > 3$ has a chord; that is there exists an edge, not of the circuit, which connects two of the circuit's vertices.

It is not straightforward to express the simple circuit notion in IFP logic. That is why we looked for some alternative chordal graph deønition. But instead of it we found out that the following algorithm that checks graph chordality (see [Gav72]) can easily be adapted in order to be expressible in IFP logic formula.

Algorithm 1 (for chordality checking)

Step 1. Search for a vertex v such that all the vertices adjacent to it are connected to each other (or form a clique).

Step 2. Eliminate v and all the edges incident to it.

Step 3. Repeat Steps 1˘2 until no such a vertex can be found.

The general problem of graph K-colorability is NP-complete but for $K = 2$ it is solvable in polynomial time. The polynomiality of the case is based on the following theorem:

Theorem 2
A graph is not 2-colorable iœ there exist a cycle of odd length in a graph.

The formula that describes the class of 2-colorable graphs is also based on the above theorem. Here is an IFP formula that says: ıthere exist a path of odd length in a graph.ȷ

$$\varphi_{\text{oddpath}}(x,y) = [\text{IFP}_{xy,X}(Exy \vee \exists vw(Xxv \wedge Evw \wedge Ewy))] \tag{11}$$

Thus, a fact that a graph is 2-colorable can be described as:

$$\varphi_{\text{2-col}} = \neg(\exists x \varphi_{\text{oddpath}}(x,x)) \tag{12}$$

Also 2-colorability problem can be expressed in TC logic.

3.4 Vertex Cover

Definition 4 (Vertex Cover)
Let $G(V,E)$ be a graph, and positive integer $K \leq |V|$. There is a vertex cover of size K or less for G, i.e., a subset $V' \subseteq V$ with $|V'| \leq K$ such that for each edge $\{u,v\} \in E$ at least one of u and v belongs to V'.

Definition 5 (Independent Set)
Independent set S is a set of vertices such that there is no edge between any pair of vertices from S.

In general, the problem of Vertex Cover is NP-complete but when V' is required to be both a vertex cover and an independent set for G it is easily solved in polynomial time.

We are going to describe the corresponding class of graphs. The following lemma states the relation between a vertex cover and an independent set.

Lemma 1
Let $G(V,E)$ be a graph, V' is a vertex cover for G iœ $V - V'$ is an independent set for G.

And the formula that determines oddness of the degree of the vertex v is:

$$\varphi_{\text{vodd}}(v) = \exists v_{\max}(Evv_{\max} \wedge \forall a(Eva \Rightarrow \neg\varphi_{\text{vpairs}}(a, v_{\max}, v))) \tag{7}$$

The formula $\varphi_{\text{veven}}(v) = \neg\varphi_{\text{vodd}}(v)$ expresses evenness of the vertex v.

Now we are ready to write in a straightforward way the formula that describes the class of Euler graphs according to the Theorem 1. φ_{con} describes graph connectivity, see formula (2).

$$\begin{aligned}\varphi_{\text{euler}} =& \varphi_{\text{con}} \wedge \Big(\exists v_1 v_2(\neg v_1 = v_2 \wedge \varphi_{\text{vodd}}(v_1) \wedge \varphi_{\text{vodd}}(v_2) \wedge \\ & \forall x(\neg x = v_1 \wedge \neg x = v_2 \Rightarrow \neg\varphi_{\text{vodd}}(x))) \vee \forall x(\neg\varphi_{\text{vodd}}(x))\Big)\end{aligned} \tag{8}$$

Note, that evenness of a ordered set and consequently the class of Euler graphs is expressible in TC logic.

The following formula builds the transitive closure of the relation Rxy, “there exists exactly one element between x and y”. The transitive closure of this relation means “find all pairs of elements such that there exist only odd number of intermediate elements”. Now, lets fix x, $x = min$ and we will obtain “all pairs of elements such that the first element is minimal and the second one is separated from it by odd number of elements”, that is if we consider $\min = 1$ then the second element can have only odd number. Thus, if the set is even then $\max$, the maximal element of the set, should be even number and will not belong to the transitive closure.

$$\varphi_{\text{odd-els}}(\min, y) = [\text{TC}_{x,y}(\exists z Sxz \wedge Szy)] \min y \tag{9}$$

$$\varphi_{\text{even}} = \neg\varphi_{\text{odd-els}}(\min, \max) \tag{10}$$

3.3 Graph 2-Colorability

Definition 3 (K-Colorability)
Let $G(V, E)$ be a graph, and positive integer $K \leq |V|$. G is K-colorable iff there exist a function $f : V \to \{1, 2, \ldots, K\}$ such that $f(u) \neq f(v)$ whenever $\{u, v\} \in E$.

(ii) $a <^A b$ or $b <^A a$

(iii) if $a <^A b$ and $b <^A c$ then $a <^A c$

(iv) $S^A ab$ iœ $(a <^A b$ and for all c, if $a <^A c$ then $b <^A c$ or $b = c)$

(v) $\min^A <^A a$ or $\min^A = a$

(vi) $a <^A \max^A$ or $a = \max^A$

The following formula accumulates pairwise in the set X successive elements of a set starting from min, the minimal element. Thus, if the set is even then max, the maximal element, will be present in the set and if the set is odd then max will not be present in the set.

$$\varphi_{\text{pairs}}(x, y) = [\text{IFP}_{xy,X}(x = \min \wedge Sxy \vee \exists z_1 z_2(Xz_1z_2 \wedge Sz_2x \wedge Sxy))] \tag{5}$$

Thus, the formula $\varphi_{\text{odd}} = \forall x \neg \varphi_{\text{pairs}}(x, \max)$ expresses oddness of the set and the formula $\varphi_{\text{even}} = \neg \varphi_{\text{odd}}$ ¯ evenness of the set.

Now, we can use the obtained formulae for set evenness/oddness in order to describe vertices degree evenness/oddness. First, let us øx an ordered τ-structure for the graph $G(V, E)$: $\tau_{\text{graph}} = \{E, <, S, min, max\}$. The idea is to look at the set of vertices that are adjacent to a given vertex as a set with its own min, max and PS, pseudo-successor relation, deøned relatively to the vertex v as follows.

(i) $\varphi_{\min}(x, v) = Evx \wedge \forall a(Eva \wedge \neg x = a \Rightarrow x < a)$

(ii) $\varphi_{\max}(x, v) = Evx \wedge \forall a(Eva \wedge \neg x = a \Rightarrow x > a)$

(iii) $\varphi_{PS}(x, y, v) = Evx \wedge Evy \wedge x < y \wedge \forall a(Eva \wedge x < a \wedge \neg y = a \Rightarrow y < a)$

Now, the formula that accumulates in the set X pairs of pseudo-successive vertices adjacent to v starting from min, looks as follows:

$$\begin{aligned} \varphi_{\text{vpairs}}(x, y, v) = &[\text{IFP}_{xyv,X}(\varphi_{\min}(x, v) \wedge \varphi_{PS}(x, y, v) \vee \\ &\exists z_1 z_2(Xz_1z_2v \wedge \varphi_{PS}(z_2, x, v) \wedge \varphi_{PS}(x, y, v)))]xyv \end{aligned} \tag{6}$$

states that for all vertices x there is no path that starts and ends in the same vertex, that is there are no circuits in the graph.

Note, that we don't need the explicit formula for a simple circuit. This is due to the fact that a simple circuit is always shorter than a not simple circuit of which it forms a part. And if a pair of vertices is connected by two paths the formula $\varphi_{\text{path}}(x, x)$ finds the shortest one. That means that if the set contains a pair (x, x) this means there is a simple circuit that starts and ends in x.

In fact the class of trees is axiomatizable in Transitive Closure Logic, since the formula $\varphi_{\text{path}}(x, y)$ is expressible in it.

$$\varphi_{\text{path}}(x, y) = [\mathrm{TC}_{x,y} Exy]xy \tag{4}$$

3.2 Euler Graph

Definition 2 (Euler Graph)
An Euler path of an undirected graph $G(V, E)$ is a path $e_1, e_2, \ldots, e_l$ such that every edge appears on it exactly once; thus, $l = |E|$. An undirected graph which has an Euler path is called an Euler graph.

It is not straightforward to express this definition in IFP formula. So, we used the following theorem that describes an Euler graph in a different way.

Theorem 1
An undirected connected graph is an Euler graph if and only if exactly two vertices are of odd degree or all vertices are of even degree. In the latter case, every Euler path of the graph is a circuit, and in former case, none is.

From the above theorem it follows that we have to express:

(i) Graph connectivity (what we already did).

(ii) Evenness/oddness of a vertex degree.

Let us first describe a more general thing, the evenness of a set. For this we will need an ordering on a set.

Let $\tau = \{<, S, \min, \max\}$, where S stands for a successor relation. A τ-structure $\mathcal{A}$ is called an ordering if for all $a, b, c \in A$:

(i) not $a <^A a$

The semantics is deøned inductively w.r.t. the calculus above, the meaning of $[\mathrm{IFP}_{\overline{x},X}\,\varphi]\overline{t}$ being $\overline{t} \in F_\infty^{(X\overline{x}\vee\varphi)}$. More precisely: If X is k-ary and the variables free in $[\mathrm{IFP}_{\overline{x},X}\,\varphi]\overline{t}$ are among $\overline{u}$ and $\overline{Y}$, and $\overline{b}$ and $\overline{S}$ are interpretations in $\mathcal{A}$ of $\overline{u}$ and $\overline{Y}$, respectively, then

$$\mathcal{A} \models [\mathrm{IFP}_{\overline{x},X}\,\varphi]\,\overline{t}\,[\overline{b},\overline{S}] \quad \text{iœ} \quad (t_1[\overline{b}],\ldots,t_k[\overline{b}]) \in F_\infty^{(X\overline{x}\vee\varphi)}$$

We sometimes denote the relation $F_\infty^{(X\overline{x}\vee\varphi)}$ by $[\mathrm{IFP}_{\overline{x},X}\,\varphi]\,\overline{t}$.

3 Some Polynomial Time Graph Properties and Corresponding FP Deønitions

We consider only ønite graphs without self-loops and parallel edges.

3.1 Tree

Definition 1 (Tree)
Let $G(V,E)$ be an undirected graph. G is called circuit-free if there are no simple circuits in G. G is called a tree if it is connected and circuit-free.

First, let us øx an unordered τ-structure for the graph $G(V,E)$: $\tau_{\text{graph}} = \{E\}$. We use straightforwardly the above deønition to describe logically a tree. That is, $\varphi_{\text{tree}} = \varphi_{\text{con}} \wedge \varphi_{\text{cf}}$, where φ_{con} describes connected graphs and φ_{cf} describes circuit-free graphs. Both φ_{con} and φ_{cf} use $\varphi_{\text{path}}(x,y)$, the formula that accumulates in the set X all pairs of vertices (x,y) connected by a path in a graph.

(1) $$\varphi_{\text{path}}(x,y) = [\mathrm{IFP}_{xy,X}(Exy \vee \exists z(Xxz \wedge Ezy))]xy$$

Now, the sentence

(2) $$\varphi_{\text{con}} = \forall xy(\neg x = y \Rightarrow \varphi_{\text{path}}(x,y))$$

says that all pairs of vertices x and y, such that $x \neq y$, are connected by a path, thus expressing graph connectivity.

And the sentence

(3) $$\varphi_{\text{cf}} = \forall x \neg\varphi_{\text{path}}(x,x)$$

and similarly for FO(DTC).

The meaning of $[\mathrm{TC}_{\overline{x},\overline{y}}\,\varphi(\overline{x},\overline{y},\overline{u})]\overline{s}\overline{t}$ is

$$(\overline{s},\overline{t}) \in \mathrm{TC}\left(\{(\overline{x},\overline{y}) \mid \varphi(\overline{x},\overline{y},\overline{u})\}\right),$$

and the meaning of $[\mathrm{DTC}_{\overline{x},\overline{y}}\,\varphi(\overline{x},\overline{y},\overline{u})]\overline{s}\overline{t}$ is deøned analogically.
(here $\{(\overline{x},\overline{y}) \mid \varphi(\overline{x},\overline{y},\overline{u})\}$ is considered as a binary relation on the set of length$(\overline{x})$-tuples of the universe).

2.2 IFP Logic

Let us ørst deøne the øxed-point F_∞. Fix a ønite set M. A function $F : \mathrm{Pow}(M) \to \mathrm{Pow}(M)$ gives rise to a sequence of sets

$$\emptyset, F(\emptyset), F(F(\emptyset)), \ldots$$

Denote its members by $F_0, F_1, \ldots$, i.e. $F_0 = \emptyset$ and $F_{n+1} = F(F_n)$. F_n is called the n-th stage of F. Suppose that there is an $n_0 \in \mathbf{N}$ such that $F_{n_0+1} = F_{n_0}$, that is, $F(F_{n_0}) = F_{n_0}$. Then $F_m = F_{n_0}$ for all $m \geq n_0$. We denote F_{n_0} by F_∞ and say that the øxed-point F_∞ of F exists. In case the øxed-point F_∞ does not exist, we agree to set $F_\infty := \emptyset$.

Now we introduce inÆationary øxed-point logic FO(IFP). It contains ørst-order logic and is closed under øxed-points of deønable inÆationary operations.

The syntax of FO(IFP): For a vocabulary τ the class of formulas of FO(IFP) of vocabulary τ is given by the calculus

- $\dfrac{}{\varphi}$ where φ is an atomic second-order formula over τ
- $\dfrac{\varphi}{\neg\varphi}$; $\dfrac{\varphi,\psi}{\varphi\vee\psi}$; $\dfrac{\varphi}{\exists x\varphi}$
- $\dfrac{\varphi}{[\mathrm{IFP}_{\overline{x},X}\,\varphi]\overline{t}}$ where the lengths of $\overline{x}$ and $\overline{t}$ are the same and coincide with the arity of X.

Sentences are formulas without free ørst-order and second-order variables, where the free occurrence of variables is deøned in the standard way, adding the clause

$$\mathrm{free}\left([\mathrm{IFP}_{\overline{x},X}\,\varphi]\overline{t}\right) := \mathrm{free}(\overline{t}) \cup \left(\mathrm{free}(\varphi)/\{\overline{x},X\}\right).$$

$K \in$ LOGSPACE	iœ	$K \in$ DTC
$K \in$ NLOGSPACE	iœ	$K \in$ TC
$K \in$ PTIME	iœ	$K \in$ IFP
$K \in$ NPTIME	iœ	$K \in \sum_1^1$
$K \in$ PSPACE	iœ	$K \in$ PFP

2.1 TC Logics

Here we present the deønition of Deterministic Transitive Closure Logic (DTC) and Transitive Closure Logic (TC).

Let R be a binary relation on a set M, $R \subseteq M^2$. The transitive closure TC(R) of R is deøned by

$$\mathrm{TC}(R) := \{(a,b) \in M^2 | \text{ there exist } n > 0 \text{ and } e_0, \dots, e_n \in M \text{ such that } a = e_0,\ b = e_n, \text{ and for all } i < n, (e_i, e_{i+1}) \in R\}.$$

And the deterministic transitive closure is deøned by

$$\mathrm{DTC}(R) := \{(a,b) \in M^2 | \text{ there exist } n > 0 \text{ and } e_0, \dots, e_n \in M \text{ such that } a = e_0,\ b = e_n, \text{ and for all } i < n, e_{i+1} \text{ is the unique } e \text{ for which } (e_i, e) \in R\}.$$

Transitive Closure Logic FO(TC) and Deterministic Transitive Closure Logic FO(DTC) are obtained by closing FO under the transitive closure and the deterministic transitive closure of deønable relations, respectively. More precisely: For a vocabulary τ the class of formulas of FO(TC) of vocabulary τ is given by the calculus

- $\frac{}{\varphi}$ where φ is an atomic ørst-order formula over τ
- $\frac{\varphi}{\neg\varphi}$; $\frac{\varphi, \psi}{\varphi \vee \psi}$; $\frac{\varphi}{\exists x \varphi}$
- $\frac{\varphi}{[\mathrm{TC}_{\overline{x},\overline{y}}\, \varphi]\overline{s}\overline{t}}$ where the variables in $\overline{x}\,\overline{y}$ are pairwise distinct and where the tuples $\overline{x}, \overline{y}, \overline{s}$ and $\overline{t}$ are all of the same length, $\overline{s}$ and $\overline{t}$ being tuples of terms.

For FO(DTC) the last rule is replaced by $\frac{\varphi}{[\mathrm{DTC}_{\overline{x},\overline{y}}\, \varphi]\overline{s}\overline{t}}$ with the same side conditions. We deøne

$$\mathrm{free}([\mathrm{TC}_{\overline{x},\overline{y}}\, \varphi]\overline{s}\overline{t}) := \mathrm{free}(\overline{s}) \cup \mathrm{free}(\overline{t}) \cup (\mathrm{free}(\varphi)/\{\overline{x}, \overline{y}\}),$$

1 Introduction

This work deals with the expressibility of some graph polynomial properties in inÆationary øxed point (IFP) logic language and it is based on that part of the Main theorem that connects PTIME complexity class with deønability in IFP logic and on the following inequality:

$$\mathrm{FO} \leq \mathrm{FO(DTC)} \leq \mathrm{FO(TC)} \leq \mathrm{FO(IFP)}$$

In fact some polynomial time properties of graphs can be expressed in FOL, or in Deterministic Transitive Closure (DTC) Logic, or in Transitive Closure (TC) Logic, thus having LOGSPACE and NLOGSPACE complexity according to the Main theorem. But we gave IFP deønitions even for those problems that can be stated in (D)TC logic. In order to express some graph properties sometimes we needed ordering on the structure, sometimes not.

1.1 Structure of the Work

In section one we recall some logic deønitions. DTC, TC and IFP logics are deøned and the Main Theorem is stated. In section two a number of graph polynomial properties are stated and IFP logic deønitions of corresponding classes of structures are given.

1.2 The Main Result of the Work

It is shown how not only some graph deønitions can be stated in IFP logic but also how some polynomial algorithms can be adapted to be expressed in IFP logic. In this way the algorithm for chordality checking was expressed, thus the class of chordal graphs is deøned in IFP logic.

2 Fixpoint Logics and Computational Complexity

Here is the Main theorem that connects computational complexity classes with diœerent logics.

Let K be a class of ordered τ-structures, $K \subseteq \mathcal{O}(\tau)$. We write $K \in \mathrm{IFP}$ if K is axiomatizable in FO(IFP), and use similar notations for the other logics.

Fixed Point Logic Descriptions of Some Graph Polynomial Properties

Victoria Skoblikov Maria Zapolotzki

September 27, 1997