

A Logical Characterisation of Linear Time on Nondeterministic Turing Machines

Clemens Lautemann, Nicole Schweikardt, Thomas Schwentick

Institut für Informatik / FB 17
Johannes Gutenberg-Universität D-55099 Mainz
email: {cl|nisch|tick}@informatik.uni-mainz.de

Abstract. The paper gives a logical characterisation of the class $\text{NTIME}(n)$ of problems that can be solved on a nondeterministic Turing machine in linear time. It is shown that a set L of strings is in this class if and only if there is a formula of the form $\exists f_1 \cdots \exists f_k \exists R_1 \cdots \exists R_m \forall x \varphi$ that is true exactly for all strings in L . In this formula the f_i are unary function symbols, the R_i are unary relation symbols and φ is a quantifier-free formula. Furthermore, the quantification of functions is restricted to *non-crossing*, *decreasing* functions and in φ no equations in which different functions occur are allowed. There are a number of variations of this statement, e.g., it holds also for $k = 3$. From these results we derive an Ehrenfeucht game characterisation of $\text{NTIME}(n)$.

1 Introduction

Since Fagin's seminal result that NP is the class of problems that can be described by an existential second-order (ESO) formula [6] there have been several characterisations of subclasses of NP by sublogics of ESO [7, 14, 9]. Lynch showed that all problems in $\text{NTIME}(n)$, i.e., all problems that can be solved in linear time on nondeterministic Turing machines, can also be expressed by a monadic ESO formula in the presence of a built-in addition relation.

Although $\text{NTIME}(n)$ is a relatively robust class, in order to capture the notion of linear time as used in algorithm design, Turing machines seem far too restrictive: apparently, simple operations, such as traversing a tree, cannot be done. Therefore, a number of alternative models have been proposed to capture this notion. Notably, in a series of papers ([9, 8, 10]), Grandjean introduced and investigated linear-time classes DLIN and NLIN, based on deterministic and nondeterministic random access machines. NLIN contains $\text{NTIME}(n)$ as a subclass but it is not known whether this inclusion is strict. Grandjean proved that Lynch's logic even captures (at least) all the languages in NLIN, hence indicating that this logic probably does not exactly characterise $\text{NTIME}(n)$. He also showed that a set L is in NLIN if and only if it is the set of models of formulas of the form $\exists f_1 \cdots \exists f_k \forall x \varphi$, where φ is quantifier-free and the second-order quantifiers $\exists f_i$ range over unary functions.

In the present paper, we give an *exact* characterisation of $\text{NTIME}(n)$. The motivation behind exact logical characterisations of (presumably) weaker and

weaker complexity classes is the hope that they might enable lower bound proofs by methods of Finite Model Theory like Ehrenfeucht games (cf. [5]). Such games have been successfully used in non-expressibility results for similar logics (for a survey see e.g. [17]).

Our characterisation is obtained by restricting Grandjean’s logic¹ for NLIN. The main difference concerns the function quantifiers. In our logic, quantification of functions is restricted to *non-crossing* functions, i.e., functions on $\{1, \dots, n\}$ whose graph, when drawn in the upper half plane with vertices on the line, has no crossing arcs. Such functions have, in different guises, been used before to describe computations, e.g., they play an important part in the lower bound proof of [15], and in the separation of $\text{DTIME}(n)$ from $\text{NTIME}(n)$, proved in [16]. In the form of *matchings*, they were used in a logical characterisation of context-free languages in [13]. In fact, we make use of the close connection between context-free languages and $\text{NTIME}(n)$, which is expressed in the theorem of [3], stating that a set is in $\text{NTIME}(n)$ iff it is the projective image of the intersection of three context-free languages.

To be more precise, we show that a set of strings can be recognised by a nondeterministic Turing machine in linear time iff it is the set of models of a formula $\exists f_1 \cdot \dots \exists f_k \exists R_1 \cdot \dots \exists R_m \forall x \varphi$, where φ is a quantifier-free formula (with certain syntactical restrictions), the second-order quantifiers $\exists R_i$ range over unary relations, and the function quantifiers $\exists f_i$ range over decreasing non-crossing functions only. By restricting the number, k , of function variables, we obtain a strict hierarchy of classes from $k = 0$ to $k = 3$: $k = 0$ characterises the regular languages [4], $k = 1$ the context-free languages, and for $k \geq 3$, we obtain $\text{NTIME}(n)$. Using the lower bound from [15], it can be seen that the class of languages defined by formulas with $k = 2$ function quantifiers lies strictly between context-free languages and $\text{NTIME}(n)$.

Our logic is fairly robust, allowing a number of variations. If we want to show that some set is contained in $\text{NTIME}(n)$, we can do this by using a rather liberal syntax. On the other hand, in order to show that some set is *not* contained in $\text{NTIME}(n)$, the more constrained our formula class, the better. We present an Ehrenfeucht game for $\text{NTIME}(n)$, based on our most restrictive characterisation, in which the players play only three rounds of rather restricted moves.

2 Preliminaries

2.1 Strings and Structures

Let Σ be an alphabet (i.e. a finite nonempty set). By ϵ we denote the *empty* string, Σ^* is the class of all finite strings over Σ , and $\Sigma^+ := \Sigma^* \setminus \{\epsilon\}$. By $|w|$ we denote the *length* of a string $w \in \Sigma^*$.

The signature τ_Σ associated to an alphabet Σ consists of two constant symbols *min* and *max*, unary function symbols s and p , and a unary relation symbol

¹ Note however, that Grandjean’s encoding of strings as structures is different from ours, which is the straightforward one.

W_σ for each letter $\sigma \in \Sigma$. With each string $w = w_1 \cdots w_n \in \Sigma^+$ we associate the τ_Σ -structure $\underline{w} := \langle [n], 1, n, s, p, (W_\sigma)_{\sigma \in \Sigma} \rangle$, where $[n]$ is an abbreviation for $\{1, \dots, n\}$, $s(i) := i+1$ for $i < n$, $s(n) := n$, $p(i) := i-1$ for $i > 1$, $p(1) := 1$, and, for every $\sigma \in \Sigma$, $W_\sigma := \{i \in [n] / w_i = \sigma\}$.

2.2 Formulas

We consider structures with unary functions, unary relations and constants. Consequently, terms are built from variables, constants, and function symbols, and an atomic formula is either a term equality or a unary relation symbol (also called a *predicate*) applied to a term. We will consistently use g, f, f_i for function symbols, R, Q, R_i, Q_i, W_σ for unary relation symbols.

Our logics will be fragments of existential second-order logic, obtained by both syntactic and semantic restrictions. If Φ is a class of formulas, we write $\forall x \Phi$ for the class of all those formulas of the form $\forall x \varphi$, where $\varphi \in \Phi$. Analogously we use notations like $\exists f_1 \cdots f_k \Phi$, $\exists f \Phi$, $\exists R_1 \cdots R_m \Phi$. Thus, as an example, $\psi \in \exists f_1 f_2 f_3 \exists \bar{R} FO$ means that ψ is of the form $\exists f_1 \exists f_2 \exists f_3 \exists R_1 \cdots \exists R_m \forall x \varphi$, for some $m \geq 0$, where φ is a first-order formula.

Our semantic restrictions concern the scope of the function quantifiers $\exists f_i$. Let, for every n , F_n be a class of functions on $[n]$, $F := \bigcup_{n \geq 1} F_n$. For a formula φ and a string w we write $\underline{w} \models^F \exists f_1 \cdots \exists f_k \varphi$ iff there are functions $f_1^w, \dots, f_k^w \in F_{|w|}$ such that $\langle \underline{w}, f_1^w, \dots, f_k^w \rangle \models \varphi$. Accordingly, if $\psi = \exists f_1 \cdots \exists f_k \varphi$ we write $\text{Mod}^F(\psi)$ for the F -model set of ψ , i.e., the set $\{w / \underline{w} \models^F \psi\}$, and for a class Ψ of formulas, $\text{MOD}^F(\Psi) := \{\text{Mod}^F(\psi) / \psi \in \Psi\}$.

2.3 Non-Crossing Functions

Let $f : [n] \rightarrow [n]$ be a *non-increasing* function, i.e., $f(j) \leq j$, for all $j \in [n]$. We call f *non-crossing*, iff for all $j, j' \in [n]$ such that $f(j) < j' \leq j$, it holds that $f(j') \geq f(j)$. Let NNC denote the class of all non-increasing, non-crossing functions, and DNC the class of all *decreasing* non-crossing functions (where additionally to $f \in \text{NNC}$ we require that $f(j) < j$, for all $j > 1$). Instead of $f \in \text{NNC}$ ($f \in \text{DNC}$) we shall also say f is *nnc* (f is *dnc*). With regard to expressive power, in our context, the difference between NNC and DNC is immaterial, as the following lemma shows.

Lemma 1. *Let $\Psi_k := \exists f_1 \cdots f_k \exists \bar{R} \forall x FO$. Then $\text{MOD}^{\text{NNC}}(\Psi_k) = \text{MOD}^{\text{DNC}}(\Psi_k)$.*

Proof. Let $\varphi = \exists f_1 \cdots \exists f_k \exists R_1 \cdots \exists R_m \forall x \chi$ be a formula in Ψ_k .

“ \supseteq ”: If we let $\varphi' := \exists f_1 \cdots \exists f_k \exists R_1 \cdots \exists R_m \forall x (\chi \wedge (x = \text{min} \vee \bigwedge_{i=1}^k f_i x \neq x))$ then, for every w , $\underline{w} \models^{\text{DNC}} \varphi$ iff $\underline{w} \models^{\text{NNC}} \varphi'$.

“ \subseteq ”: We represent a function $f_i \in \text{NNC}$ by a function $f'_i \in \text{DNC}$ together with a set I_i as follows:

$$I_i(j) \leftrightarrow f_i(j) = j, \quad \text{and} \quad f'_i(j) = \begin{cases} f_i(j) & \text{if } f_i(j) \neq j \\ p(j) & \text{if } f_i(j) = j. \end{cases}$$

We now define the formula $\varphi' := \exists f'_1 \dots \exists f'_k \exists I_1 \dots \exists I_k \exists R_1 \dots \exists R_m \forall x \chi'$, where χ' is obtained from χ in the following way: For $i \in [k]$, for any $t \in \{s, p, f_1, \dots, f_k\}^*$, and any terms y, z , where y does not contain the symbol f_i , do the following:

- Replace $Q(tf_i y)$ by $(I_i(y) \rightarrow Q(ty)) \wedge (\neg I_i(y) \rightarrow Q(tf'_i y))$,
- replace $tf_i y = z$ by $(I_i(y) \rightarrow ty = z) \wedge (\neg I_i(y) \rightarrow tf'_i y = z)$,
- replace $z = tf_i y$ in an analogous way.

Successive application leads to a formula χ' which does not contain the symbol f_i . By induction one can easily verify for all strings $w \in \Sigma^+$ that $\underline{w} \models^{NNC} \varphi$ iff $\underline{w} \models^{DNC} \varphi'$. \square

Remark 1. In both constructions in the proof of Lemma 1, the first-order quantifier structure of φ remains unchanged. Furthermore, in the first part of the proof, φ' contains the same atoms as φ , plus the equations $x = \min$ and $f_i x = x$, whereas in the second part, every term equation in φ' can be obtained from one in φ by replacing each f_i with either ϵ or f'_i .

Lemma 1 allows us to use either class, depending on which suits our purposes best. The more restrictive class *DNC* has some particularly useful properties.

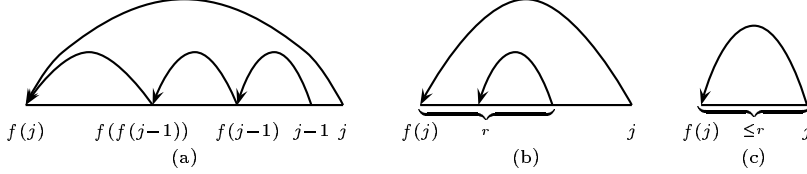


Fig. 1. (a) illustrates Lemma 2. (b), (c) illustrate the two possible cases of Lemma 3.

Lemma 2. *A function $f : [n] \rightarrow [n]$ is dnc iff $f(1) = 1$, and for every $j > 1$ there is a number $q \geq 0$ such that $f(j) = f^q(j-1)$.* Figure 1 (a) gives an illustration.

Proof. “ \Rightarrow ”: Let $n > 1$, f be dnc on $[n]$, and $j > 1$. As f is decreasing we have $f(j) \leq j-1 < j$ and the sequence $j-1, f(j-1), f^2(j-1), \dots$ is strictly decreasing (unless 1 is reached), hence there is a minimal q such that $f^q(j-1) \leq f(j)$. If $q = 0$ then we are done, otherwise $q > 0$ and $f(j) < f^{q-1}(j-1) \leq j$. Since f is non-crossing it follows that $f^q(j-1) = f(f^{q-1}(j-1)) \geq f(j)$, hence $f^q(j-1) = f(j)$.

“ \Leftarrow ”: An easy induction shows that $f(j) < j$, for all $j > 1$. We show the non-crossing property of f , i.e., $f(j) < j' \leq j \Rightarrow f(j') \geq f(j)$, by induction on $j - j'$. The base case, $j = j'$, is trivial, so let $j' < j$. Choose $q \geq 0$ such that $f^q(j-1) = f(j)$, and let $q_0 < q$ be maximal with $f^{q_0}(j-1) \geq j'$. By induction, since $f^{q_0}(j-1) - j' < j - j'$, we have $f(j') \geq f(f^{q_0}(j-1)) \geq f^q(j-1) = f(j)$. \square

Later we will use the non-crossing property in a special form which we state here as a lemma.

Lemma 3. *Let $f : [n] \rightarrow [n]$ be dnc, and let $r > 0$. For each $j \in [n]$, there is some l , $0 \leq l < r$, such that $f(f(j)+r) = f(j)+l$, or $j = f(j)+l$. The two possible cases are illustrated in Figure 1 (b) and (c).*

Proof. We have $f(j) < f(j)+r$, and if $f(j)+r \leq j$ then $f(j) \leq f(f(j)+r) < f(j)+r$, hence $f(f(j)+r) = f(j)+l$, for some l , $0 \leq l < r$. If $f(j)+r > j$ then, clearly, $j = f(j)+l$, for some $l < r$. \square

3 The Main Result

Let Φ_k be the class of quantifier-free formulas over the signature τ_Σ which have as free variables at most one *FO* variable x , at most k unary function symbols f_1, \dots, f_k , and an arbitrary number of unary relation symbols, and in which no term equation contains occurrences of more than one of the symbols f_1, \dots, f_k . Let $\Phi_* = \bigcup_{k \geq 0} \Phi_k$.

Theorem 1. $\text{NTIME}(n) = \text{MOD}^{\text{DNC}}(\exists \bar{f} \exists \bar{R} \forall x \Phi_*)$.

The proof is given in the next subsections. We first show in Subsection 3.1 that every context-free language is the *DNC*-model set of a formula in $\exists \bar{f} \exists \bar{R} \forall x \Phi_1$. Together with the fact that every language in $\text{NTIME}(n)$ is the image under a projection of the intersection of three context-free languages (c.f., [3]) this proves the inclusion from left to right (even with three function variables), see Proposition 2. For the other direction, in Subsection 3.2 (Proposition 3), we first transform every formula in $\exists \bar{f} \exists \bar{R} \forall x \Phi_*$ into one whose atoms are all of a very restricted form, without changing its *DNC*-model set. Finally, in Subsection 3.3 (Proposition 4), we construct, for every such formula, a nondeterministic, linear time Turing machine which evaluates the formula on its input string.

3.1 Expressing Derivations

Concerning context-free grammars, we use the standard notation of [11]. Let us briefly recall the notion of a *derivation tree* for a context-free grammar $G = (V, \Sigma, P, S)$. Every $\sigma \in \Sigma$ is a derivation tree for $\sigma \Rightarrow_G^* \sigma$. If $\alpha_i \in \Sigma \cup V$, $A \rightarrow \alpha_1 \cdots \alpha_r$ is a production in P , and T_i is a derivation tree for $\alpha_i \Rightarrow_G^* u_i \in \Sigma^+$, then $A(T_1, \dots, T_r)$ is a derivation tree for $A \Rightarrow_G^* u_1 \cdots u_r$, where by $A(T_1, \dots, T_r)$ we denote the ordered tree whose root is labelled with A and has r subtrees T_1, \dots, T_r .

Proposition 1. *If a language $L \subseteq \Sigma^+$ is context-free, then $L = \text{Mod}^{\text{DNC}}(\psi)$ for a τ_Σ -formula $\psi \in \exists \bar{f} \exists \bar{R} \forall x \Phi_1$.*

Proof. As the constructions of the proof of Lemma 1 do not essentially change the structure of the FO-formulas, it suffices to find a formula ψ such that $L = \text{Mod}^{\text{NNC}}(\psi)$. Let L be generated by some grammar $G = (V, \Sigma, P, S)$ in *quadratic double Greibach normal form*, i.e., by a grammar whose productions are of the form $A \rightarrow \alpha$ where $\alpha \in \Sigma \cup \Sigma \Sigma \cup \Sigma V \Sigma \cup \Sigma V V \Sigma$ (cf. [2], Theorem 3.3). We have to find a formula ψ such that for all $w \in \Sigma^+$ we have $\underline{w} \models^{\text{NNC}} \psi$ iff $S \Rightarrow_G^* w$.

Let $w \in L$ be of length n , and let T be a derivation tree for $S \Rightarrow_G^* w$. As G is in quadratic double Greibach normal form, the leftmost and the rightmost child of each internal node of T are leaves, labelled with terminal symbols; and every position in w corresponds to a leftmost or to a rightmost child of an internal node of T . We can thus represent T by an nnc-function f^T and sets $Q_{A \rightarrow \alpha}^T$ for each production $A \rightarrow \alpha$ in P as follows: For all $j \in [n]$ we define

$$f^T(j) := \begin{cases} i & \text{if } i \text{ corresponds to the leftmost, } j \text{ to the rightmost} \\ & \text{child of the same internal node of } T, \\ j & \text{if no such } i \text{ exists.} \end{cases}$$

$$Q_{A \rightarrow \alpha}^T(j) \iff j \text{ corresponds to the rightmost child of an internal node of } T \text{ which is associated to the production } A \rightarrow \alpha, \text{ i.e., which is labelled with } A, \text{ has exactly } |\alpha| \text{ children, the } i\text{th of which is labelled with the } i\text{th letter in } \alpha \text{ (for all } 1 \leq i \leq |\alpha| \text{)}.$$

As one can easily see, $\langle \underline{w}, f^T, (Q_{A \rightarrow \alpha}^T)_{(A \rightarrow \alpha) \in P} \rangle$ satisfies the formula

$$\begin{aligned} \varphi_{tree} &:= \forall x \varphi_{disjoint} \wedge \varphi_{start} \wedge \bigwedge_{(A \rightarrow \alpha) \in P} (Q_{A \rightarrow \alpha}(x) \rightarrow \varphi_{A \rightarrow \alpha}), \text{ where} \\ \varphi_{disjoint} &:= \bigwedge_{q' \neq q \in P} \neg(Q_q(x) \wedge Q_{q'}(x)) \\ \varphi_{start} &:= x = \text{max} \rightarrow (fx = \text{min} \wedge \bigvee_{(S \rightarrow \alpha) \in P} Q_{S \rightarrow \alpha}(x)) \\ \varphi_{A \rightarrow \sigma} &:= W_\sigma(x) \wedge fx = x \\ \varphi_{A \rightarrow \sigma\tau} &:= W_\sigma(fx) \wedge W_\tau(x) \wedge fx = px \neq x \\ \varphi_{A \rightarrow \sigma B\tau} &:= W_\sigma(fx) \wedge W_\tau(x) \wedge fx = pfpfx \neq fpfx \wedge \bigvee_{(B \rightarrow \beta) \in P} Q_{(B \rightarrow \beta)}(px) \\ \varphi_{A \rightarrow \sigma C B\tau} &:= W_\sigma(fx) \wedge W_\tau(x) \wedge fx = pfpfpfx \neq fpfpfx \wedge \\ &\quad \left(\bigvee_{(B \rightarrow \beta) \in P} Q_{B \rightarrow \beta}(px) \right) \wedge \left(\bigvee_{(C \rightarrow \gamma) \in P} Q_{C \rightarrow \gamma}(pfppx) \right). \end{aligned}$$

We thus obtain $\underline{w} \models^{\text{NNC}} \psi$, where $\psi := \exists f (\exists Q_q)_{q \in P} \varphi_{tree}$. For the opposite direction let w be a string of length n , let f be an nnc-function on $[n]$, and let $(Q_{A \rightarrow \alpha})_{(A \rightarrow \alpha) \in P}$ be subsets of $[n]$ such that φ_{tree} is satisfied by $\langle \underline{w}, f, (Q_{A \rightarrow \alpha})_{(A \rightarrow \alpha) \in P} \rangle$. We have to show that $S \Rightarrow_G^* w$, i.e., that $w \in L$. For all $j \in [n]$ such that $Q_{A \rightarrow \alpha}(j)$ for some $(A \rightarrow \alpha) \in P$ we define a tree $T(j)$ as follows:

$$\begin{aligned} \text{If } Q_{A \rightarrow \sigma}(j) & \text{ then } T(j) := A(\sigma), \\ \text{if } Q_{A \rightarrow \sigma\tau}(j) & \text{ then } T(j) := A(\sigma, \tau), \\ \text{if } Q_{A \rightarrow \sigma B\tau}(j) & \text{ then } T(j) := A(\sigma, T(p(j)), \tau), \\ \text{if } Q_{A \rightarrow \sigma C B\tau}(j) & \text{ then } T(j) := A(\sigma, T(pfp(j)), T(p(j)), \tau). \end{aligned}$$

By a straightforward induction on the depth of $T(j)$ one can easily show that if $Q_{A \rightarrow \alpha}(j)$, then $T(j)$ is a derivation tree for $A \Rightarrow_G^* w_{f(j)} \cdots w_j$. As φ_{start} guarantees that $f(n) = 1$ and that $Q_{S \rightarrow \alpha}(n)$ for some production $S \rightarrow \alpha$ in P , we conclude that $T(n)$ is a derivation tree for $S \Rightarrow_G^* w$, and hence $w \in L$. \square

Let Σ and $\tilde{\Sigma}$ be alphabets. A mapping $h : \tilde{\Sigma} \rightarrow \Sigma$ is called a *projection*. We extend h to map strings in the canonical way: $h(w_1 \cdots w_n) := h(w_1) \cdots h(w_n)$. If $L \subseteq \tilde{\Sigma}^*$ then the set $h(L) := \{h(w) / w \in L\}$ is called the *projection* of L under h .²

Theorem 2 ([3], Theorem 4.1). *A language L is in $\text{NTIME}(n)$ if and only if L is a projection of the intersection of three context-free languages.*

Hence, from Proposition 1 we obtain the following:

Proposition 2. *If a language $L \subseteq \Sigma^+$ is in $\text{NTIME}(n)$, then $L = \text{Mod}^{DNC}(\psi)$ for a τ_Σ -formula $\psi \in \exists f_1 f_2 f_3 \exists \bar{R} \forall x \Phi_3$.*

Proof. From Theorem 2 we obtain an alphabet $\tilde{\Sigma}$, context-free languages $L_1, L_2, L_3 \subseteq \tilde{\Sigma}^+$, and a projection $h : \tilde{\Sigma} \rightarrow \Sigma$ such that $L = h(L_1 \cap L_2 \cap L_3)$. W.l.o.g., $\Sigma \cap \tilde{\Sigma} = \emptyset$. Proposition 1 provides formulas $\psi_i \in \exists f \exists \bar{R} \forall x \Phi_1$ over the signature $\tau_{\tilde{\Sigma}}$ such that $L_i = \text{Mod}^{DNC}(\psi_i)$ for $i \in \{1, 2, 3\}$. We define a formula

$$\psi := (\exists W_{\tilde{\sigma}})_{\tilde{\sigma} \in \tilde{\Sigma}} (\psi_1 \wedge \psi_2 \wedge \psi_3) \wedge (\forall x \bigvee_{\tilde{\sigma} \in \tilde{\Sigma}} (W_{\tilde{\sigma}}(x) \wedge W_{h(\tilde{\sigma})}(x) \wedge \bigwedge_{\tilde{\sigma}' \neq \tilde{\sigma}} \neg W_{\tilde{\sigma}'}(x))),$$

which holds for a string $w \in \Sigma^+$ iff there exists a string $\tilde{w} \in \tilde{\Sigma}^+$ of the same length, such that \tilde{w} satisfies ψ_1, ψ_2 , and ψ_3 , and $h(\tilde{w}) = w$. Hence we obtain $\text{Mod}^{DNC}(\psi) = h(L_1 \cap L_2 \cap L_3) = L$. Furthermore, ψ can easily be transformed into a formula in $\exists f_1 f_2 f_3 \exists \bar{R} \forall x \Phi_3$. \square

3.2 Simplifying Formulas

In this subsection we will prove the following proposition.

Proposition 3. *For every formula $\psi \in \exists f_1 \cdots f_k \exists \bar{R} \forall x \Phi_k$ there is a formula $\psi' \in \exists f_1 \cdots f_k \exists \bar{R} \forall x \Phi_k$ such that $\text{Mod}^{DNC}(\psi') = \text{Mod}^{DNC}(\psi)$, and the atoms of ψ' are of the following forms:*

- $x = \min, x = \max, f_i x = f_i^q p x$ (where $i \in [k]$ and $q \geq 0$),
- $Q(x), Q(sx), Q(px), Q(f_i x)$ (where $i \in [k]$ and Q is a unary relation symbol).

Furthermore, the symbol s does not occur in ψ' if it does not occur in ψ .

The proof proceeds in several steps:

² In the literature, projections are sometimes called *length-preserving homomorphisms*.

1. We replace every equational atom $tx=t'x$ by a new relational atom $Q_{\{t,t'\}}(x)$ (with the intended meaning that $Q_{\{t,t'\}}(j) \leftrightarrow t(j) = t'(j)$), and we replace $t\mu = y$ (for $\mu \in \{min, max\}$ and an arbitrary term y) by the new relational atom $Q_{t\mu}(y)$ (with the intended meaning that $Q_{t\mu}(j) \leftrightarrow t(\mu) = j$).
2. We eliminate all predicates $Q_{\{t,t'\}}$, where $\{t,t'\} \neq \{f_i, f_i^q p\}$ by replacing them with adequate formulas. (In this step we make essential use of the special properties of dnc-functions).
3. We take the conjunction of the resulting formula with formulas that define the predicates $Q_{\{f_i, f_i^q p\}}$ and $Q_{t\mu}$. After this step, all equations of the resulting formula are of the desired form.
4. We replace every relational atom $Q(tx)$ by a new relational atom $Q_t(x)$ (with the intended meaning that $Q_t(j) \leftrightarrow Q(t(j))$) and take the conjunction of the resulting formula with formulas that define the predicates Q_t .
5. A similar construction is performed to replace atoms of the form $Q(tmin)$ and $Q(tmax)$.

The details of this construction are given in the following lemmas.

We use $\text{fun}(\psi)$ for the set of all function symbols, and $\text{rel}(\psi)$ for the set of all unary relation symbols which occur in atoms of ψ .

Lemma 4. *For every formula $\psi \in \forall x \Phi_k$ there is a formula $\psi' \in \exists \bar{R} \forall x \Phi_k$ whose atoms are the equational atoms of ψ , $x=min$, $x=max$, and relational atoms of the form $Q(x)$ and $Q(gx)$ (where $g \in \{s, p, f_1, \dots, f_k\}$), such that for all $w \in \Sigma^+$, all functions f_1, \dots, f_k and all sets Q_1, \dots, Q_m , we have*

$$\langle \underline{w}, f_1, \dots, f_k, Q_1, \dots, Q_m \rangle \models \psi \quad \text{iff} \quad \langle \underline{w}, f_1, \dots, f_k, Q_1, \dots, Q_m \rangle \models \psi'.$$

Furthermore, the symbol s does not occur in ψ' if it does not occur in ψ .

Proof. Let $\psi := \forall x \varphi$. Let T be the set of terms occurring in relational atoms of ψ , and let T' be the set of all proper prefixes of elements in T , i.e., T' is a finite subset of $\{s, p, f_1, \dots, f_k\}^*$. For every $Q \in \text{rel}(\psi)$ and every $t \in T'$ we introduce a new set variable Q_t which, speaking informally, shall have the property that $Q_t(x) \leftrightarrow Q(tx)$. We define $\psi'' := (\exists Q_t)_{\substack{Q \in \text{rel}(\varphi) \\ t \in T'}} \forall x \varphi'' \wedge \chi$, where

$$\chi := \bigwedge_{Q \in \text{rel}(\psi)} ((Q_\epsilon(x) \leftrightarrow Q(x)) \wedge (\bigwedge_{\substack{tg \in T' \\ g \in \text{fun}(\psi)}} Q_{tg}(x) \leftrightarrow Q_t(gx)))$$

asserts that $Q_t(x) \leftrightarrow Q(tx)$, and φ'' is obtained from φ by replacing every occurrence of $Q(ty)$ by $Q_t(y)$ (for $y \in \{x, min, max\}$).

It remains to eliminate atoms of the form $Q(min)$, $Q(max)$. For each $Q \in \text{rel}(\psi')$ we introduce new set variables MIN_Q and MAX_Q which we intend to be interpreted with the empty set unless $Q(min)$ (resp. $Q(max)$) is true, in which case MIN_Q (resp. MAX_Q) is to be interpreted with the whole universe of the underlying structure. This interpretation can be enforced by the formula $\varphi_{MIN_Q} := (MIN_Q(x) \leftrightarrow MIN_Q(px)) \wedge (x=min \rightarrow (MIN_Q(x) \leftrightarrow Q(x)))$ and by an analogous formula φ_{MAX_Q} . We define $\psi' :=$

$$(\exists MIN_Q \exists MAX_Q)_{Q \in \text{rel}(\psi'')} (\exists Q_t)_{\substack{Q \in \text{rel}(\varphi) \\ t \in T'}} \forall x (\varphi' \wedge \chi \wedge \bigwedge_{Q \in \text{rel}(\psi')} (\varphi_{MIN_Q} \wedge \varphi_{MAX_Q})),$$

where φ' is obtained from φ'' by replacing every occurrence of $Q(\min)$ (resp. $Q(\max)$) by $MIN_Q(x)$ (resp. by $MAX_Q(x)$). As one can easily see, ψ' has the desired properties. \square

Let $w \in \Sigma^+$ be a string of length n , and let f_1, \dots, f_k be functions over $[n]$. For all $t, t' \in \{s, p, f_1, \dots, f_k\}^*$ we define the following sets: $\widehat{Q}_{all} := [n]$, $\widehat{Q}_{tmin} := \{t(1)\}$, $\widehat{Q}_{tmax} := \{t(n)\}$, and $\widehat{Q}_{\{t, t'\}} := \{j \in [n] \mid t(j) = t'(j)\}$.

Obviously, for the formulas $\varphi_{\{f_i, f_i^q p\}} := \forall x Q_{\{f_i, f_i^q p\}}(x) \leftrightarrow (f_i x = f_i^q p x)$ and $\varphi_{all} := \forall x Q_{all}(x)$ we have

$$\begin{aligned} \langle \underline{w}, f_1, \dots, f_k, Q_{all} \rangle \models \varphi_{all} & \quad \text{iff} \quad Q_{all} = \widehat{Q}_{all}, \text{ and} \\ \langle \underline{w}, f_1, \dots, f_k, Q_{\{f_i, f_i^q p\}} \rangle \models \varphi_{\{f_i, f_i^q p\}} & \quad \text{iff} \quad Q_{\{f_i, f_i^q p\}} = \widehat{Q}_{\{f_i, f_i^q p\}}. \end{aligned}$$

The following lemma shows that the predicates \widehat{Q}_{tmin} and \widehat{Q}_{tmax} can be defined in a similar way.

Lemma 5. *Let $t \in \{s, p, f_1, \dots, f_k\}^*$ and $\mu \in \{\min, \max\}$. There is a formula $\varphi_{t\mu} \in \overline{\exists R} \forall x \Phi_k$ which has no equational atoms except $x = \min$ and $x = \max$, such that for all $w \in \Sigma^+$, all functions f_1, \dots, f_k , and all sets $Q_{t\mu}$ we have $\langle \underline{w}, f_1, \dots, f_k, Q_{t\mu} \rangle \models \varphi_{t\mu}$ iff $Q_{t\mu} = \widehat{Q}_{t\mu}$.*

Furthermore, the symbol s does not occur in $\varphi_{t\mu}$ if it does not occur in t .

Proof. For $\mu \in \{\min, \max\}$, the formula $\chi_\mu := \forall x (Q_\mu(x) \leftrightarrow x = \mu)$ defines the predicate \widehat{Q}_μ . The formula $\chi_{<t\mu} :=$

$$\forall x (Q_{<t\mu}(x) \rightarrow Q_{<t\mu}(px)) \wedge (Q_\mu(x) \rightarrow (\neg Q_{<t\mu}(tx) \wedge (Q_{<t\mu}(px) \vee Q_{\min}(tx))))$$

expresses that $Q_{<t\mu}$ is closed under predecessor and that $t(\mu)$ is the smallest element not in $Q_{<t\mu}$. Hence $\chi_{<t\mu}$ defines the predicate $\widehat{Q}_{<t\mu}$ consisting of all elements smaller than $t(\mu)$.

The formula $\chi_{t\mu} := \forall x (Q_{t\mu}(x) \leftrightarrow (\neg Q_{<t\mu}(x) \wedge (Q_{<t\mu}(px) \vee Q_{\min}(x))))$ defines the predicate $\widehat{Q}_{t\mu}$. All in all, the formula

$$\varphi_{t\mu} := \exists Q_{\min} \exists Q_{\max} \exists Q_{<t\mu} (\chi_{\min} \wedge \chi_{\max} \wedge \chi_{<t\mu} \wedge \chi_{t\mu})$$

has the desired properties. \square

In the following lemma, we will make essential use of the properties of dnc-functions proved in Subsection 2.3. Hence, the condition that f_1, \dots, f_k are dnc-functions is very important.

Lemma 6. *For every formula $\varphi \in \Phi_k$ there is a formula $\varphi' \in \Phi_k$ which has no equational atoms, such that for all $w \in \Sigma^+$, all dnc-functions f_1, \dots, f_k , all sets Q_1, \dots, Q_m , and all positions j in w , we have*

$$\langle \underline{w}, f_1, \dots, f_k, Q_1, \dots, Q_m, j \rangle \models \varphi \quad \text{iff} \quad \langle \underline{w}, f_1, \dots, f_k, Q_1, \dots, Q_m, j, (\widehat{Q}_i)_{i \in I} \rangle \models \varphi',$$

where $I := \{all, t\mu, \{f_i, f_i^q p\} \mid q \geq 0, i \in [k], t \in \{s, p, f_i\}^*, \mu \in \{\min, \max\}\}$. Furthermore, the symbol s does not occur in φ' if it does not occur in φ .

Proof. We proceed in three steps:

Step 1: In the formula φ we replace every equational atom $tx=t'x$ by the relational atom $Q_{\{t,t'\}}(x)$, and we replace $t\mu=y$ (for $\mu \in \{min, max\}$ and an arbitrary term y) by the relational atom $Q_{t\mu}(y)$. This leads to a formula φ_1 which has no equational atoms. Provided that the predicates $Q_{\{t,t'\}}$ and $Q_{t\mu}$ are interpreted in their intended meaning, φ_1 is equivalent to φ .

From the definition of the class Φ_k we know that the formula φ_1 only contains predicates $Q_{\{t,t'\}}$ where $t, t' \in \{s, p, f_i\}^*$ for some $i \in [k]$. In the following two steps we show how to eliminate those predicates where $\{t, t'\} \neq \{f_i, f_i^q p\}$. We will use y as an abbreviation for an arbitrary term.

Step 2: Elimination of predicates $Q_{\{t,t'\}}$ where the symbol s occurs in t, t' :

$$\text{Since } ps(j) = \begin{cases} p(j) & \text{if } j = n \\ j & \text{else} \end{cases} \quad \text{and} \quad sp(j) = \begin{cases} s(j) & \text{if } j = 1 \\ j & \text{else} \end{cases}$$

we can replace each occurrence of

$$\begin{aligned} & - Q_{\{t_1 p s t_2, t'\}}(y) \text{ by } (Q_{max}(t_2 y) \wedge Q_{t_1 p max}(t' y)) \vee (\neg Q_{max}(t_2 y) \wedge Q_{\{t_1 t_2, t'\}}(y)), \\ & - Q_{\{t_1 s p t_2, t'\}}(y) \text{ in an analogous way,} \\ & - Q_{\{t s, t'\}}(y) \text{ by } (Q_{max}(y) \wedge Q_{t max}(t' y)) \vee (\neg Q_{max}(y) \wedge Q_{\{t, t' p\}}(s y)), \\ & - Q_{\{s t, t'\}}(y) \text{ by } (Q_{max}(t y) \wedge Q_{max}(t' y)) \vee (\neg Q_{max}(t y) \wedge \neg Q_{min}(t' y) \wedge Q_{\{t, p t'\}}(y)). \end{aligned}$$

For a dnc-function f , from Lemma 3 we know that for every $r > 0$ and every j there is some $0 \leq l < r$ such that $j = s^l f(j)$ or $f s^r f(j) = s^l f(j)$. Hence, for each $f \in \{f_1, \dots, f_k\}$, we can replace each occurrence of $Q_{\{t_1 f s^r f t_2, t'\}}(y)$ by

$$\bigvee_{i=0}^{r-1} \left\{ (Q_{\{\epsilon, s^i f\}}(t_2 y) \wedge Q_{\{t_1 f s^{r-i}, t'\}}(y)) \vee (Q_{\{\epsilon, f s^{r-i}\}}(s^i f t_2 y) \wedge Q_{\{t_1 s^i f t_2, t'\}}(y)) \right\}.$$

Note that in the above replacement rules each predicate $Q_{\{t,t'\}}$ is replaced by a formula using only predicates $Q_{\{\tilde{t}, \tilde{t}'\}}$ where $\tilde{t}\tilde{t}'$ contains fewer occurrences of the symbol s or $\tilde{t}\tilde{t}'$ is shorter than tt' and does not contain more occurrences of the symbol s . Hence, successive application of these replacement rules transforms φ_1 into a formula φ_2 which contains only predicates $Q_{\{t,t'\}}$ where the symbol s does not occur in t, t' , and which is equivalent to φ_1 if each f_i is interpreted by a dnc-function.

Step 3: Elimination of predicates $Q_{\{t,t'\}}$ where $\{t, t'\} \neq \{f_i, f_i^q p\}$:

Note that for all predicates $Q_{\{t,t'\}}$ occurring in φ_2 we have $t, t' \in \{p, f_i\}^*$ for some $i \in [k]$.

As the equation $y=y$ is always true, we can replace $Q_{\{\epsilon, \epsilon\}}(y)$ by $Q_{all}(y)$. Since f_i and p are decreasing, if $t \neq \epsilon$, then $ty=y$ is true iff $y=min$. Hence, we can replace $Q_{\{t, \epsilon\}}(y)$ by $Q_{min}(y)$. Obviously, we can replace $Q_{\{\tilde{t}, t'\tilde{t}\}}(y)$ by $Q_{\{t,t'\}}(\tilde{t}y)$.

We now explain how to replace predicates of the form $Q_{\{t f, t' p\}}(y)$ (where $f \in \{f_1, \dots, f_k\}$ and $\{t f, t' p\} \neq \{f, f^q p\}$). Since f is dnc, for every $j \in [n]$ we know that $f^{|t'|} p(j) \leq t' p(j)$. On the other hand, by Lemma 2 there is some $q \geq 0$ such that $f^q p(j) = f(j)$. Hence $t f(j) = t' p(j)$ iff there is some $q \in \{0, \dots, |t'|\}$ such

that $tf^q p(j) = t'p(j)$. We can therefore replace every occurrence of $Q_{\{tf, t'p\}}(y)$ by

$$\bigvee_{q=0}^{|t'|} (Q_{\{f, f^q p\}}(y) \wedge Q_{\{tf^q, t'\}}(py)).$$

Note that in the above replacement rules each predicate $Q_{\{t, t'\}}$ (where $\{t, t'\} \neq \{f_i, f_i^q p\}$) is replaced by a formula using predicates $Q_{\{\tilde{t}, \tilde{t}'\}}$ where $\tilde{t}\tilde{t}'$ contains fewer occurrences of the symbol p or $\tilde{t}\tilde{t}'$ is shorter than tt' and does not contain more occurrences of the symbol p . Hence, successive application of these replacement rules transforms φ_2 into an equivalent formula φ_3 which contains only predicates $Q_{\{t, t'\}}$ where $\{t, t'\} = \{f_i, f_i^q p\}$ for some $q \geq 0$. From the arguments above, it should be clear that the formula $\varphi' := \varphi_3$ has the desired properties. \square

Lemma 7. *For every formula $\psi \in \forall x \Phi_k$ there is a formula $\psi' \in \exists \bar{R} \forall x \Phi_k$ whose atoms are of the form $x = \min$, $x = \max$, $f_i x = f_i^q p x$, $Q(x)$, and $Q(gx)$ (where $g \in \{s, p, f_1, \dots, f_k\}$, $i \in [k]$, and $q \geq 0$), such that for all $w \in \Sigma^+$, all dnc-functions f_1, \dots, f_k , and all sets Q_1, \dots, Q_m we have*

$$\langle \underline{w}, f_1, \dots, f_k, Q_1, \dots, Q_m \rangle \models \psi \quad \text{iff} \quad \langle \underline{w}, f_1, \dots, f_k, Q_1, \dots, Q_m \rangle \models \psi'.$$

Furthermore, the symbol s does not occur in ψ' if it does not occur in ψ .

Proof. Let $\psi := \forall x \varphi$. Let φ' be the formula from Lemma 6, let T be the (finite) set of all $t \in \{s, p, f_1, \dots, f_k\}^*$ for which a predicate $Q_{t \min}$ or $Q_{t \max}$ occurs in φ' , and let r be maximal such that a predicate $Q_{\{f_i, f_i^q p\}}$ occurs in φ' . We define the formula

$$\begin{aligned} \psi' := & \exists Q_{all} \exists (Q_{\{f_i, f_i^q p\}})_{\substack{i \in [k], \\ 0 \leq q \leq r}} (\exists Q_{t \min} \exists Q_{t \max})_{t \in T} \\ & (\forall x \varphi') \wedge \varphi_{all} \wedge \bigwedge_{\substack{i \in [k], \\ 0 \leq q \leq r}} \varphi_{\{f_i, f_i^q p\}} \wedge \bigwedge_{t \in T} (\varphi_{t \min} \wedge \varphi_{t \max}). \end{aligned}$$

From the lemmas 6 and 5, we conclude that the equational atoms of ψ' are of the desired form, and that for all $w \in \Sigma^+$, all dnc-functions f_1, \dots, f_k , and all sets Q_1, \dots, Q_m we have $\langle \underline{w}, f_1, \dots, f_k, Q_1, \dots, Q_m \rangle \models \psi$ iff $\langle \underline{w}, f_1, \dots, f_k, Q_1, \dots, Q_m \rangle \models \psi'$. Lemma 4 allows us to transform ψ' into a formula which has the desired properties. \square

Note that Lemma 7 directly implies Proposition 3.

3.3 Evaluating Formulas

We now conclude the proof of Theorem 1 by showing how a nondeterministic Turing machine can evaluate formulas of the form given in Proposition 3.

Proposition 4. *Let ψ be of the form $\exists f_1 \dots \exists f_k \exists R_1 \dots \exists R_m \forall x \varphi$, where φ is a quantifier-free formula in which all atoms are of one of the following forms:*

- $x=min, x=max, f_i x=f_i^q p x$ (where $i \in [k]$ and $q \geq 0$),
- $Q(x), Q(sx), Q(px), Q(f_i x)$ (where $i \in [k]$ and Q is a unary relation symbol).

Then there is a nondeterministic, linear time Turing machine which accepts precisely those strings w for which $\underline{w} \models^{DNC} \psi$.

Proof. The machine, M , will scan its input from left to right, guessing the values of all the relations, and evaluating φ accordingly. Since writing down (and reading) the values $f_i(j)$ would take too long ($\Omega(\lg n)$ steps per input position) M represents these values indirectly, by the movements of pushdown heads. To this end, we equip M with k pushdown tapes, one for each of the function variables f_i . When scanning the input at position j , pushdown tape i will hold information about $f_i(j), f_i^2(j), \dots, 1$, in this order.

More precisely, on input w , M proceeds in $n = |w|$ "metasteps", where in the j th metastep it looks at the j th input symbol w_j . It maintains three variables, P^-, P, P^+ , where $P^- = (\langle r_1^-, \dots, r_m^- \rangle, \sigma^-)$ contains the information about the previous position, $j-1$. r_l^- is intended to be the truth value of R_l , σ^- the input symbol, w_{j-1} , at that position. P and P^+ contain the same information about the current and the next position, respectively. We will also write $P(j)$ when referring to the value of P at position j . For each $j \in [n]$, at the beginning of the j th metastep, M is scanning input symbol w_{j-1} . It then does the following:

1. Move the input head to the right.
2. Set $P^- := P, P := P^+$.
3. Update the pushdown tapes (in order for the i th pushdown to show $P(f_i(j))$ as topmost entry). This is done by performing one of two possible activities:
 - (a) Push(P^-), set $q := 0$. For $j = 1$, this is M 's only option.
(This corresponds to $f_i(j)=j-1$.)
 - (b) Pop an arbitrary number, $q' \geq 0$, of entries, but not all entries, set $q := q'+1$. (This corresponds to $f_i(j)=f_i^{q'}(j-1)$.) In this case, M remembers whether or not $q \leq r_i$, where r_i is the largest number r for which φ contains the atom $f_i x=f_i^r p x$, and if so, M stores q , otherwise it sets $q := \infty$.
4. If $j = n$, set $P^+ := P$, otherwise guess a pair $P^+ = (\langle r_1^+, \dots, r_m^+ \rangle, \sigma^+)$.

At this point, M 's input head scans w_j , P^-, P and P^+ contain the truth values of the guessed relations, and about the input symbol at positions $j-1, j$, and $j+1$, respectively. For each $i \leq k$ the i th pushdown tape stores a sequence which is intended to be $P(f_i^q(j-1)), P(f_i^{q+1}(j-1)), \dots, P(1)$. Let $P^i := (\langle r_1^i, \dots, r_m^i \rangle, \sigma^i)$ be the top entry of the i th pushdown tape.

Now M proceeds by evaluating all atoms for $x = j$.

5. Reject if $\sigma \neq w_j$.
6. Evaluate all atoms of the form $x=min$, or $x=max$.
7. Evaluate all atoms of the form $W_{\sigma'}(z)$:
 $W_{\sigma'}(x) = \text{true iff } \sigma' = \sigma, W_{\sigma'}(sx) = \text{true iff } \sigma' = \sigma^+, W_{\sigma'}(px) = \text{true iff } \sigma' = \sigma^-, W_{\sigma'}(f_i x) = \text{true iff } \sigma' = \sigma^i$.

8. Evaluate all atoms of the form $R_l(z)$:
 $R_l(x) = r_l, R_l(sx) = r_l^+, R_l(px) = r_l^-, R_l(f_i x) = r_l^i$.
9. Evaluate all atoms of the form $f_i x = f_i^r p x$:
 If $r < q$, evaluate $f_i x = f_i^r p x$ to false, if $r = q$, to true. If $r > q$, then the value is true iff P^i is the only entry left on the i th pushdown tape (i.e., $P^i = P(1)$).

If, with these values, φ evaluates to false, M rejects.

To begin with, M 's input head scans the tape square to the left of the input, and before performing the first metastep, M guesses P^+ and sets $P := P^+$.

Clearly, each of these steps, except for step 3, can be performed in constant time. Since in each metastep, at most one pair P is stored on each pushdown tape, during the whole computation no more than n pairs can be popped, so in total, M only needs $O(n)$ steps.

We now have to show that M accepts an input string w if and only if $\underline{w} \models^{DNC} \psi$. Let $f_1, \dots, f_k \in DNC$, and $R_1, \dots, R_m \subseteq [n]$ be chosen in such a way that $\langle \underline{w}, f_1, \dots, f_k, R_1, \dots, R_m \rangle \models \forall x \varphi$. Then, at each position $j \in \{0, \dots, n-1\}$, M can guess the values P^+ in accordance with w and R_1, \dots, R_m . Assume that in metastep j , M moves its i th pushdown head according to 3a, if $f_i(j) = j-1$, and according to 3b, if $f_i(j) = f_i^q(j-1)$ for some $q > 0$. It is easily seen by induction that in metastep j , the pushdown tape contains the sequence $P(f_i(j)), P(f_i^2(j)), \dots, P(1)$. From this it follows that M accepts its input.

For the other direction, assume that M accepts the string w . Let $R_1, \dots, R_m \subseteq [n]$ be as defined by the values r_l , guessed by M . Define the functions f_1, \dots, f_k according to the movements of the pushdown heads: if, in metastep j , M pushes onto the i th tape, set $f_i(j) := j-1$. If M pops q' elements, set $f_i(j) = f_i^{q'+1}(j-1)$. By Lemma 2 these functions are then dnc, and since M accepts, it follows that the evaluation of $\varphi(x)$ yields true, at each input position j . Therefore $\langle \underline{w}, f_1, \dots, f_k, R_1, \dots, R_m \rangle \models \forall x \varphi$, i.e., $\underline{w} \models^{DNC} \psi$. \square

Remark 2. It should be noted that in the proof of Proposition 4 M scans its input only once, from left to right, and uses as many pushdown tapes as ψ has function variables. In particular, for $k = 1$, the formula can be evaluated by a pushdown automaton, i.e., the language is context-free.

4 Discussion and Further Results

Like complete problems, a logical characterisation of a complexity class can expose what is typical for that class. Our characterisation shows that, in some sense, non-crossing functions capture the essence of linear time on nondeterministic Turing machines. $NTIME(n)$ allows some variation in the precise definition of Turing machines. This is reflected in the logic: we can vary both our syntactical and semantical restrictions quite considerably, without changing the class of model sets. Some of these variations lead to interesting insights: we obtain a strict hierarchy of classes within $NTIME(n)$ and a characterisation of the class by an Ehrenfeucht game, which looks considerably easier to play for the duplicator than the game one obtains directly from Theorem 1.

4.1 Variations

The only restriction on formulas in $\exists \bar{f} \exists \bar{R} \forall x \Phi_*$ is, that no term equation contains occurrences of more than one of the symbols f_1, \dots, f_k . We have hence characterised $\text{NTIME}(n)$ by formulas of a rather liberal syntax.

In the proofs of the propositions 1 and 2, the symbol s is not needed. On the other hand, given the function s , the predecessor function p can be existentially quantified (since p is dnc) and defined by a formula in $\forall x \Phi_1$. Hence, it is not essential that both symbols, s and p , belong to the signature of strings — one of them would suffice.

The unary relation symbols may be omitted in the logic for $\text{NTIME}(n)$, because one unary relation R can be encoded by two dnc-functions as follows: We use one dnc-function f where $f(j) = p(j)$ if $j \in R$ and $f(j) = fp(j)$ otherwise. Clearly, $p(j) \neq fp(j)$ unless $j = 1, 2$. A second dnc-function, f' , can be used to encode the membership of $j = 1, 2$, e.g., by $f'(j+2) = p(j+2)$ if $j \in R$, and $f'(j+2) = f'p(j+2)$ otherwise.

On the other hand, provided that we may use an arbitrary number of unary relations, from Proposition 2 we know that already *three* dnc-functions suffice for describing languages in $\text{NTIME}(n)$; and according to Proposition 3, for the first-order part of the describing formula, only atoms of a very restricted form are necessary.

From Lemma 1 and Remark 1 we know that the function class DNC can be replaced by the function class NNC . Let us mention that Theorem 1 is also valid if we replace DNC by the class PNC of all non-crossing *permutations* (which are not necessarily non-increasing).

We obtain the following proposition, where by Φ'_k we denote the class of all Boolean combinations of atoms of the forms $x = \min$, $x = \max$, $f_i x = x$, $f_i x = f_i^q p x$, $Q(x)$, $Q(gx)$, for $g \in \{s, p, f_1, \dots, f_k\}$ and unary relation symbols Q .

Proposition 5. *For $F \in \{DNC, NNC, PNC\}$ we have*

$$\text{NTIME}(n) = \text{MOD}^F(\exists \bar{f} \forall x \Phi_*) = \text{MOD}^F(\exists \bar{f} \exists \bar{R} \forall x \Phi_*) = \text{MOD}^F(\exists f_1 f_2 f_3 \exists \bar{R} \forall x \Phi'_3).$$

From the results of [13] one obtains a different logic for $\text{NTIME}(n)$. There, over the relational signature $\{<, W_\sigma / \sigma \in \Sigma\}$, it was shown that³ $\text{CFL} = \text{MOD}^{\text{MATCH}}(\exists M \text{FO})$, where MATCH is essentially the class of all injective, partial dnc-functions. Together with the theorem of Book and Greibach (Theorem 2), it follows that a language L is in $\text{NTIME}(n)$ iff $L = \text{Mod}^{\text{MATCH}}(\psi)$ for a formula $\psi := \exists M_1 \exists M_2 \exists M_3 \exists R_1 \dots \exists R_m (\varphi_1 \wedge \varphi_2 \wedge \varphi_3)$, where $<$ and M_i are the only binary relation symbols occurring in the FO -formula φ_i (for $i \in \{1, 2, 3\}$).

4.2 Separations

As stated in Remark 2, the number of function symbols corresponds to the number of pushdown tapes needed for evaluating a formula (with respect to the

³ By CFL we denote the class of all context-free languages.

function class $F \in \{DNC, NNC\}$). As a consequence, we obtain a strict hierarchy of classes from $k=0$ to $k=3$ by restricting the number k of function variables, as illustrated in the following picture. The class $MOD^F(\exists f_1 f_2 \exists \overline{R} \forall x \Phi_2)$ is separated from CFL by the language $\{ww / w \in \{0, 1\}^+\}$, and $NTIME(n)$ is separated from $MOD^F(\exists f_1 f_2 \exists \overline{R} \forall x \Phi_2)$ by the result of [15], which says that the language *SMT* (*sparse matrix transposition*) cannot be accepted with two pushdown tapes in time $o(n \log n)$, but with three pushdown tapes it can (even deterministically) be accepted in linear time.

$$\begin{array}{rcc}
MOD^F(\exists \overline{f} \exists \overline{R} \forall x \Phi_*) & = & MOD^F(\exists f_1 f_2 f_3 \exists \overline{R} \forall x \Phi_3) = NTIME(n) \\
& & \downarrow \\
& & MOD^F(\exists f_1 f_2 \exists \overline{R} \forall x \Phi_2) \\
& & \downarrow \\
& & MOD^F(\exists f \exists \overline{R} \forall x \Phi_1) = CFL \\
& & \downarrow \\
& & MOD^F(\exists \overline{R} \forall x \Phi_0) = REG
\end{array}$$

4.3 Games

Ehrenfeucht games have been proved a useful tool for showing inexpressibility results in Finite Model Theory. For a general introduction to Ehrenfeucht games we refer to the textbooks of Immerman [12] and of Ebbinghaus and Flum [5]. Our game for $NTIME(n)$ makes use of the idea of Ajtai and Fagin [1] that in Ehrenfeucht games for existential second-order logic the duplicator can choose the second structure after the spoiler has selected relations (or, in our case, functions) for the first structure. From each of the possible logical characterisations of $NTIME(n)$ one can derive a corresponding Ehrenfeucht game. We are going to describe here one variant that looks particularly easy to play for the duplicator. In particular, after choosing functions and colourings, both players have to select only one position in each string. A closer inspection of the proof of Proposition 1 shows that in the characterisation of $NTIME(n)$ we can restrict ourselves to *special* nnc-functions f which, apart from being nnc, have the following properties:

- $f(f(j)) = f(j)$ for all $j \in [n]$,
- for every j there is at most one $j' \neq j$ with $f(j') = j$, i.e., f is one-one, if we ignore loops $f(j) = j$, and
- the *width* of each arc $(f(j), j)$ is at most 2, where by *width* we denote the number of arcs lying on the surface beneath that arc. To be precise, $(f(j), j)$ has width 0 if $f(j) = j$ or $j-1$, width 1 if $f(j) = f(j-1)-1$, and width 2 if $f(j) = f(f(j-1)-1)-1$.

The game for a set L of strings consists of the following three rounds:

1. The spoiler chooses a number $m \geq 0$. Afterwards, the duplicator chooses a string $w \in L$. In the following, let n denote the length of w .
2. The spoiler chooses *special* nnc-functions f_1, f_2, f_3 on $[n]$, and colours w with m colours. Afterwards, the duplicator chooses a string $w' \notin L$, *special* nnc-functions f'_1, f'_2, f'_3 on $[n']$, and m colours on w' . (Here, n' denotes the length of w' .)

3. The spoiler chooses $j' \in [n']$. Afterwards, the duplicator chooses $j \in [n]$.

The duplicator wins the game iff the following conditions are satisfied:

- $j = 1$ iff $j' = 1$, and $j = n$ iff $j' = n'$,
- the colour of position j in w is the same as the colour of position j' in w' . The corresponding statements hold for the positions $j-1$ and $j'-1$, and for $f_i(j)$ and $f'_i(j')$ (for $i \in \{1, 2, 3\}$).
- The arc $(f_i(j), j)$ has the same width as the arc $(f'_i(j'), j')$, and it is a loop iff $(f'_i(j'), j')$ is a loop (for $i \in \{1, 2, 3\}$).

Proposition 6. *A set L of strings is in $\text{NTIME}(n)$ iff the spoiler has a winning strategy in the game for L .*

Acknowledgements We would like to thank Malika More and Arnaud Durand for stimulating discussions.

References

1. M. Ajtai and R. Fagin. Reachability is harder for directed than for undirected finite graphs. *Journal of Symbolic Logic*, 55(1):113–150, 1990.
2. J. Autebert, J. Berstel, and L. Boasson. Context-free languages and pushdown automata. *Handbook of Formal Languages*, 2:111–174, 1997.
3. R. Book and S. Greibach. Quasi-realtime languages. *Math. Syst. Theory*, 4:97–111, 1970.
4. J.R. Büchi. Weak second-order arithmetic and finite automata. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 6:66–92, 1960.
5. H.D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer-Verlag, New York, 1995.
6. R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In R.M. Karp, editor, *Complexity of Computation*, volume 7 of *SIAM-Proceedings*, pages 43–73. AMS, 1974.
7. E. Grädel. Capturing complexity classes by fragments of second order logic. *Theoretical Computer Science*, 101:35–57, 1992.
8. E. Grandjean. Invariance properties of RAMs and linear time. *Computational Complexity*, 4:62–106, 1994.
9. E. Grandjean. Linear time algorithms and NP-complete problems. *SIAM Journal of Computing*, 23:573–597, 1994.
10. E. Grandjean. Sorting, linear time and the satisfiability problem. *Annals of Mathematics and Artificial Intelligence*, 16:183–236, 1996.
11. J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley Publishing Company, 1979.
12. N. Immerman. *Descriptive and Computational Complexity*. Springer-Verlag, New York, 1998.
13. C. Lautemann, T. Schwentick, and D. Thérien. Logics for context-free languages. In *Proceedings of the Annual Conference of the EACSL*, volume 933 of *Lecture Notes in Computer Science*, pages 205–216, 1994.
14. J. F. Lynch. The quantifier structure of sentences that characterize nondeterministic time complexity. *Computational Complexity*, 2:40–66, 1992.

15. W. Maass, G. Schnitger, E. Szemerédi, and G. Turán. Two tapes versus one for off-line turing machines. *Computational Complexity*, 3:392–401, 1993.
16. W. Paul, N. Pippenger, E. Szemerédi, and W. Trotter. On determinism versus nondeterminism and related problems. In *Proc. 24th Ann. Symp. Found. Comput. Sci.*, pages 429–438, 1983.
17. T. Schwentick. Padding and the expressive power of existential second-order logics. In *11th Annual Conference of the EACSL, CSL '97*, pages 461–477, 1997.