Bounded Fixed-Point Definability and Tabular Recognition of Languages

Hans Leiß

Centrum für Informations-
und Sprachverarbeitung
Universität München
Wagmüllerstr. 23
D-80538 München
leiss@cis.uni-muenchen.de

Abstract. By relating positive inductive definitions to space-bounded computations of alternating Turing machines, Rounds, *Comp. Linguistics 14, 1988*, has given uniform grammatical characterizations of the *EXPTIME* and *PTIME* languages. But his proof gives fairly poor bounds for language recognition with context-free resp. head grammars.
We improve Rounds' analysis in two respects: first, we introduce a modified class of language definitions that allow restricted forms of negative inductions, and second, we show how to build table-driven recognizers from such definitions. For a wide and natural class of language definitions we thereby obtain fairly efficient recognizers; we can recognize the boolean closure of context-free resp. head languages in the well-known $O(n^3)$ resp. $O(n^6)$ steps on a *RAM*. Our 'bounded' fixed-point formulas apparently can not define an arbitrary *PTIME* language.
Our method is based on the existence of fixed-points for a class of operators that need neither be monotone nor increasing, but assume a norm or at least a well-founded quasi-ordering on the underlying set.

1 Introduction

Vardi[14] and Immerman[7] have shown that languages $L \in PTIME$ are those that can be defined by a formula φ in a first-order relational language (with ordering) extended by a least-fixed-point operator:

$$L = L(\varphi) := \{\, w \in \Sigma^* \mid \mathcal{A}_w \models \varphi(0, |w|)\,\}, \tag{1}$$

where a word $w = a_1 \cdots a_{|w|}$ over a finite alphabet Σ is seen as the finite relational structure

$$\mathcal{A}_w := (|w| + 1, +, \cdot, <, R_a)_{a \in \Sigma}, \tag{2}$$

an initial segment of the natural numbers $\mathcal{N} = (I\!N, +, \cdot, <)$ (modulo $|w| + 1$), expanded by relations R_a between positions connected by a in w. Rounds[12] has given a characterization of the*EXPTIME* languages in the same spirit, implicitly using 'initial segments' of the monoid $\mathcal{L} = (\Sigma^*, \cdot, a)_{a \in \Sigma}$, i.e. finite models

$$\mathcal{L}^{\leq |w|} = (\Sigma^{\leq |w|}, \cdot, a)_{a \in \Sigma},$$

where $\Sigma^{\leq n}$ is the set of words over Σ whose length is at most n. Moreover, Rounds shows that both the *PTIME* and his *EXPTIME* characterization have a uniform proof: a least fixed-point formula φ corresponds to a space-bounded alternating Turing machine $\mathcal{M}_\varphi$ (modified to cover fixed-points). For φ in the arithmetical language, a binary representation of numbers (for positions in w) leads to a $log(n)$-space-bounded machine $\mathcal{M}_\varphi$, hence

$$L(\varphi) = L(\mathcal{M}_\varphi) \in \mathit{ASPACE}(\log n) = \mathit{PTIME},$$

while for φ in the language of concatenation, one gets an n-space-bounded machine $\mathcal{M}_\varphi$, and hence

$$L(\varphi) = L(\mathcal{M}_\varphi) \in \mathit{ASPACE}(n) = \mathit{EXPTIME}.$$

In both cases, the positive inductive definition φ leads to a complexity bound for recognition of $L(\varphi)$ with deterministic Turing machines: in the *PTIME* case, the number of configurations of the corresponding *ATM* $\mathcal{M}_\varphi$ is $O(|w|^{p+3})$ where $p := |\mathit{free}(\varphi)|+|\mathit{bound}(\varphi)|$, and hence, by a result of Chandra e.a.[1], a simulation of $\mathcal{M}_\varphi$ by a *DTM* can be done in $O(|w|^{2(p+3)})$ steps.

Rounds counts $p = 3$ for context-free grammars and hence obtains a *DTM*-recognition algorithm of time complexity $O(|w|^{12})$ for context-free languages. This is far worse than the well-known algorithms by Cocke, Younger, Kasami, and Earley that do it in $O(|w|^3)$ steps on a *RAM* or even a Turing machine (see Harrison[4], p.437 ff). A similar defect of $O(n^{18})$ versus a known bound of $O(n^6)$ (cf. Joshi and Vijay-Shanker[15]) resulted in the case of head grammars, a class of grammars studied in theoretical linguistics (c.f. Section 5.1). [1]

Our aim was to understand why the method yields very poor bounds in these cases and what improvement on an abstract level could be made in reading off the recognition complexity from an inductive language definition. Since the best known recognition algorithms use a well-formed substring table to store intermediate results, our second aim was to find a logical characterization of 'languages that admit a tabular parser'.

First, we observe that the monotonicity of the induction behind least-fixed-point definitions of context-free languages is not essential – neither for the existence of fixed points nor for the efficiency of language recognition. Instead, we use a fixed-point construction for generally non-monotone, 'bounded' operators on sets with a reflexive transitive relation $\leq$ where $<$ is well-founded.

Second, we give a syntactic characterization of first-order formulas $\varphi(x, S)$ that define such operators and are invariant under going from structures $\mathcal{A}$ to 'local' substructures $\mathcal{A}^{\leq a}$ consisting of all elements $b \leq a$ in $\mathcal{A}$, in the sense that

$$\mathcal{A} \models \varphi(a, B) \iff \mathcal{A}^{\leq a} \models \varphi(a, \{\, b \in B \mid b < a \,\}).$$

Interpreting these 'bounded' formulas over $\mathcal{L}$ and exploiting more closely the syntactic form of context-free grammars, the construction of recognition tables

[1] Rounds' $p = 3$ is the *quantifier depth* of φ for a context-free grammar G in Chomsky normal form, but the *number* p of bound individual quantifiers of φ is $3n$, if G has n nonterminals A with a branching rule $A \to B\,C$. Thus, one only obtains a bound of $O(|w|^{2(3n+3)})$ for context-free and $O(|w|^{2(6n+3)})$ for head-languages.

and the staging of a bounded induction on certain finite substructures $\mathcal{L}^{\leq w}$ of $\mathcal{L}$ turn out to be essentially the same.

The efficiency of the tabular recognizers of Cocke e.a. depends on two parameters: the size of the table and the effort to compute a new table entry from given ones, which reflects a sharing of subcomputations by using stored results.

Restricting a positive induction over $\mathcal{L}$ to $\mathcal{L}^{\leq |w|}$ would in general define a language in *EXPTIME*. To obtain small recognition tables, we interprete bounded inductive definitions in the smaller finite structures

$$\mathcal{L}^{\leq w} = (\Sigma^{\leq w}, \cdot, a)_{a \in \Sigma},$$

where $\Sigma^{\leq w}$ is the set of subwords of w: for intuitively context-free language definitions $\varphi(x)$, we expect the global reading of φ in the infinite structure $\mathcal{L}$ to coincide with its local readings in the finite structures $\mathcal{L}^{\leq w}$, i.e.

$$\{\, w \in \Sigma^* \mid \mathcal{L} \models \varphi(w)\,\} = \{\, w \in \Sigma^* \mid \mathcal{L}^{\leq w} \models \varphi(w)\,\}. \tag{3}$$

This reflects, we think, the proper *logical notion*[2] *of context-independence*: grammatical properties of a string depend only on the grammatical properties of its substrings. All our bounded fixed-point formulas satisfy (3) and define operators Γ which reach their fixed point in $O(|w|^2)$ many stages of constant size, corresponding to a small recognition table with $O(|w|^2)$ fields.

The efficiency of computing a table entry is related to a peculiarity of individual quantification in the language definition that has been overlooked in Rounds' analysis. Not only do context-free or head grammars just quantify over *sub*strings of the input; more restrictively, they *decompose* it into segments of non-overlapping consecutive substrings. Restricting individual quantifiers accordingly, we introduce a class of 'decomposition grammars' that contain the boolean closure of context free ones. For these the tabular recognizers yield $O(n^3)$ recognition algorithms on a *RAM*, since computing a field can be done in $O(|w|)$ steps.

Head grammars, which define languages of splitted strings – i.e. binary relations between strings –, are similar to context-free grammars, except that concatenation is replaced by a number of 'head wrapping' operations. We generalize these to a class of operations on m-tuples of strings. Decomposition grammars using these operations define languages of strings with m segments, and the table-recognizers we obtain yield $O(n^{3m})$ recognition algorithms; a subclass has recently been introduced by Hotz and Pitsch[6]. In particular, a language in the boolean closure of the head languages is recognized in $O(n^6)$ steps.

2 Non-Monotone Operators with Fixed Points

For easier comparison we recall the Tarski/Knaster-fixed-point construction for monotone operators. An operator $\Gamma : 2^A \to 2^A$ is *monotone*, if $\Gamma(S) \subseteq \Gamma(T)$

[2] The *technical* notion of context-freeness can be expressed in second-order logic, cf. Lautemann and Schwentick[8], or by regular fixed-point expressions, cf. Leiß[9]. We remark that structures $\mathcal{A}_w$ and binary second order quantifieres are used in [8], but with structures $\mathcal{L}^{\leq w}$ one can use fixed-point formulas of monadic second order logic.

whenever $S \subseteq T \subseteq A$, and Γ is *increasing* (or *inflationary*, see Gurevich and Shelah[3]), if $S \subseteq \Gamma(S)$ for each $S \subseteq A$.

Theorem 1 (Tarski/Knaster). *If $\Gamma : 2^A \to 2^A$ is monotone or increasing, then Γ has a distinguished fixed point $\Gamma^\infty \subseteq A$, defined in stages $\Gamma^{<\alpha} \subseteq A$ by*

$$\Gamma^\infty := \bigcup \{\, \Gamma(\Gamma^{<\alpha}) \mid \alpha \text{ an ordinal} \}, \quad \Gamma^{<\alpha} := \bigcup \{\, \Gamma(\Gamma^{<\beta}) \mid \beta < \alpha \,\}. \qquad (4)$$

Recall that Γ^∞ is the least fixed point if Γ is monotone, but not in general.

A *norm* on a set S is a function $|\cdot| : S \to \eta$ onto an ordinal η. Each monotone operator Γ on A gives rise to a norm on its fixed point Γ^∞ by associating to each $a \in \Gamma^\infty$ the least ordinal α such that $a \in \Gamma(\Gamma^{<\alpha})$. Conversely, in situations where a norm or just a well-founded transitive relation $<$ on the universe is given, there are additional (definable) operators that do have fixed points.

Definition 2. Let $(A, \leq)$ be a quasi-ordering, i.e. a set A with a reflexive and transitive relation $\leq$. For $a, b \in A$, define $a < b :\iff a \leq b \wedge b \not\leq a$ and $a \sim b :\iff a \leq b \wedge b \leq a$, and for $S \subseteq A$ use

$$S^{\leq a} := \{\, b \in S \mid b \leq a \,\}, \; S^{<a} := \{\, b \in S \mid b < a \,\}, \; S^a := \{\, b \in S \mid b \sim a \,\}. \; (5)$$

We call $\leq$ a *well-founded quasi-ordering on A*, if $\leq$ is a quasi-ordering and $<$ is well-founded. An operator $\Gamma := (\Gamma_1, \ldots, \Gamma_n)$ with $\Gamma_i : 2^A \times \cdots \times 2^A \to 2^A$ is called *$<$-bounded*, if for each Γ_i, each $a \in A$ and all sets $S_1, \ldots, S_n \subseteq A$,

$$a \in \Gamma_i(S_1, \ldots, S_n) \iff a \in \Gamma_i(S_1^{<a}, \ldots, S_n^{<a}).$$

Γ is *norm-bounded*, if $(A, \leq)$ is given by a norm $|\cdot| : A \to \eta$, with $a \leq b$ iff $|a| \leq_{On} |b|$.

Example 1. If $A = \{a, b, c\}$ with $a < b \not\leq c$ and $a < c \not\leq b$, the operator $\Gamma(S) :=$ **if** $a \in S$ **then** $\{b\}$ **else** $\{c\}$ is $<$-bounded, but neither monotone nor increasing.

The relations $\leq$ and $<$ are invariant under the equivalence $\sim$, and $S^{<a}$, $S^{\leq a}$ and S^a depend on the equivalence class $[a]$ of a only. We write $S^{<|a|}$ etc. when $\leq$ on A comes from a norm. Well-foundedness of $<$ holds trivially in all finite structures. In the unary case, Γ is $<$-bounded iff $\Gamma(S)^a = \Gamma(S^{<a})^a$ for all $a \in A$ and $S \subseteq A$, whence $\Gamma(S) = \bigcup \{\, \Gamma(S^{<a})^a \mid a \in A \,\}$ for all $S \subseteq A$.

Theorem 3. *Each $<$-bounded operator $\Gamma := (\Gamma_1, \ldots, \Gamma_n) : (2^A)^n \to (2^A)^n$ on a well-founded quasi-ordering $(A, \leq)$ has a unique fixed-point $\Gamma^\infty = (\Gamma_1^\infty, \ldots, \Gamma_n^\infty)$, where the Γ_i^∞ are simultaneously defined by*

$$\begin{aligned} \Gamma_i^\infty &:= \bigcup \{\, \Gamma_i(\Gamma_1^{<a}, \ldots, \Gamma_n^{<a})^a \mid a \in A \,\} \quad \text{and} \\ \Gamma_i^{<a} &:= \bigcup \{\, \Gamma_i(\Gamma_1^{<b}, \ldots, \Gamma_n^{<b})^b \mid b < a \,\}. \end{aligned}$$

In fact,

$$\begin{aligned} \Gamma_i(\Gamma_1^{<a}, \ldots, \Gamma_n^{<a})^a &= \{\, b \sim a \mid b \in \Gamma_i(\Gamma_1^{<b}, \ldots, \Gamma_n^{<b}) \,\}, \\ \Gamma_i^\infty &= \{\, b \in A \mid b \in \Gamma_i(\Gamma_1^{<b}, \ldots, \Gamma_n^{<b}) \,\}. \end{aligned}$$

The difference between the subset $S^{<a}$ of $S \subseteq A$ and the $\Gamma^{<a} = (\Gamma_1^{<a}, \ldots, \Gamma_n^{<a})$ obtained from an operator Γ should be clear from the context.

Proof. Consider the unary case. Since $<$ is well-founded, $\Gamma^{<a}$ is well-defined. Note that $a \sim b$ implies $\Gamma^{<a} = \Gamma^{<b}$ and so $\Gamma(\Gamma^{<a})^a = \Gamma(\Gamma^{<b})^b$. We get

$$\Gamma(\Gamma^{<a})^a = \{\, b \sim a \mid b \in \Gamma(\Gamma^{<a}) \,\} = \{\, b \sim a \mid b \in \Gamma(\Gamma^{<b}) \,\},$$

from which the characterization for Γ^∞ follows. For each $a \in A$ we have

$$\begin{aligned} (\Gamma^\infty)^a = \Gamma^\infty \cap [a] \;&=\; \bigcup \{\, \Gamma(\Gamma^{<b}) \cap [b] \cap [a] \mid b \in A \,\} \\ = \bigcup \{\, \Gamma(\Gamma^{<b})^b \mid b \sim a \,\} \;&=\; \Gamma(\Gamma^{<a})^a, \qquad \text{and so} \end{aligned}$$

$$(\Gamma^\infty)^{<a} = \bigcup \{\, (\Gamma^\infty)^b \mid b < a \,\} = \bigcup \{\, \Gamma(\Gamma^{<b})^b \mid b < a \,\} = \Gamma^{<a}.$$

Putting these together, we obtain that Γ^∞ is a fixed point of Γ:

$$\begin{aligned} \Gamma^\infty = \bigcup \{\, \Gamma(\Gamma^{<a})^a \mid a \in A \,\} \;&=\; \bigcup \{\, \Gamma((\Gamma^\infty)^{<a})^a \mid a \in A \,\} \\ = \bigcup \{\, \Gamma(\Gamma^\infty)^a \mid a \in A \,\} \;&=\; \Gamma(\Gamma^\infty), \end{aligned}$$

using the $<$-boundedness of Γ in the third step. If $S = \Gamma(S)$ is another fixed point, then $S^b = \Gamma(S)^b = \Gamma(S^{<b})^b$ for each b, and by well-foundedness of $<$ one gets $\Gamma^{<a} = S^{<a}$ for each $a \in A$. This gives $\Gamma^\infty = \bigcup \{\, \Gamma(S^{<a})^a \mid a \in A \,\} = S$.

We still have $\Gamma^{<a} \subseteq \Gamma^{<b}$ for $a \leq b$. The stages are similar to the stages $\Gamma^{<\alpha} = \bigcup \{\, \Gamma(\Gamma^{<\beta}) \mid \beta < \alpha \,\}$ of Tarski's construction for monotone operators. The basic difference is that from $\Gamma(\Gamma^{<b})$ we only select the $c \sim b$, while for elements $c \not\sim b$, membership in Γ^∞ is fixed at other stages. As with monotone operators, nested recursions can be transformed into simultaneous ones:

Lemma 4. *Let $\Gamma, \Delta : 2^A \times 2^A \to 2^A$ be $<$-bounded on the well-founded quasi-ordering $(A, \leq)$. Then $\Delta_S(T) := \Delta(S, T)$ and $\Theta(S) := \Gamma(S, \Delta_S^\infty)$ are $<$-bounded operators on 2^A, and Θ^∞ is the first component Γ^∞ of the fixed-point of the simultaneous $<$-bounded operator (Γ, Δ).*

The class of $<$-bounded operators can be extended somewhat, without loosing the existence of fixed points. Call $\Gamma : 2^A \to 2^A$ *$\leq$-bounded*, if for each $a \in A$ and $S \subseteq A$, $a \in \Gamma(S)$ iff $a \in \Gamma(S^{\leq a})$. Fixed points can no longer be constructed via

$$\Gamma^{\leq a} := \bigcup \{\, \Gamma(\Gamma^{\leq b})^b \mid b \leq a \,\},$$

as this would not be well-defined. We have to insist that $\Gamma(S)^a$ monotonically depends on S^a. We call Γ *locally monotone*, if $\Gamma(S^{\leq a})^a \subseteq \Gamma(T^{\leq a})^a$ for each $a \in A$ and $S, T \subseteq A$ such that $S^{<a} = T^{<a}$ and $S^a \subseteq T^a$.

Theorem 5. *If Γ is $\leq$-bounded and locally monotone on a well-founded quasi-ordering $(A, \leq)$, then Γ has a fixed point Γ^∞, which is defined using*

$$\begin{aligned} \Gamma^\infty &:= \bigcup \{\, \Gamma(\Gamma^{\leq a})^a \mid a \in A \,\}, & \Gamma^{<a} &:= \bigcup \{\, \Gamma(\Gamma^{\leq b})^b \mid b < a \,\}, \\ \Gamma^{\leq a} &:= \bigcup \{\, \Gamma_a^{<\beta} \mid \beta \in On \,\}, & \Gamma_a^{<\beta} &:= \Gamma^{<a} \cup \bigcup \{\, \Gamma(\Gamma_a^{<\gamma})^a \mid \gamma < \beta \,\}, \end{aligned}$$

for elements $a \in A$ and ordinals β. In fact, $\Gamma^\infty = \{\, a \in A \mid a \in \Gamma(\Gamma^{\leq a}) \,\}$.

3 Definable Bounded Fixed-Point Operators

To apply the fixed-point constructions of the previous section on first-order structures $\mathcal{A}$ with a norm $|\cdot| : A \to \eta$, or a well-founded quasi-ordering $\leq$ on A, we will consider formulas $\varphi(x, S)$ with a free set variable S such that

$$\mathcal{A} \models \forall S \forall x\, (\, \varphi(x,S) \leftrightarrow \varphi(x, S^{<|x|})\,), \text{ resp. } \mathcal{A} \models \forall S \forall x\, (\, \varphi(x,S) \leftrightarrow \varphi(x, S^{<x})\,).$$

Each such formula defines a norm- resp. <-bounded operator

$$\Gamma_\varphi(S) := \{\, a \in A \mid \mathcal{A} \models \varphi(a, S)\,\}$$

on A whose fixed-point Γ_φ^∞ is taken as the meaning of a new predicate $\mu S \lambda x \varphi$. Our intended application is the structure $\mathcal{A} = \mathcal{L}$ of strings over a finite alphabet Σ, with $|\cdot|$ being the length of strings and $\leq$ the substring relation.

Definition 6. *Concatenation bounded fixed-point formulas* over Σ, or *CBFP-formulas*, are given by

$$\begin{aligned} \varphi :\equiv\ & x = a \mid x_1 = x_2 \mid x_1 = x_2 \cdot x_3 \mid S(x) && (a \in \Sigma) \\ & \mid \neg\varphi_1 \mid (\varphi_1 \vee \varphi_2) \mid \exists x_1 < x_2\ \varphi && (x_1 \not\equiv x_2) \\ & \mid \mu(S_1, \ldots, S_n)(\lambda x_1\, \varphi_1, \ldots, \lambda x_n\, \varphi_n)(x), \end{aligned}$$

where in the last clause, the set variables S_i and individual variables x_i are pairwise distinct, $\mathit{freeIndV}(\varphi_i) \subseteq \{\, x_i\,\}$, and no φ_i contains an atomic subformula $S(x_i)$ with a set variable S (not necessarily among $S_1, \ldots, S_n$).

Remark. Officially, we consider $x = a$ and $x = y \cdot z$ as syntactic sugar for atomic formulas $a(x)$ and $\mathit{Cat}(x, y, z)$ in a *relational* language. We use $x = \epsilon$ for the formula saying that x is neither a letter nor composed of strict substrings.

Our 'bounded fixed point' formulas are different from those of 'bounded fixed point logic' as described in Ebbinghaus and Flum[2], Section 7.7.

Definition 7. Let $\mathcal{L} = (\Sigma^*, \cdot, \{a\})_{a \in \Sigma}$ be the set of all finite strings over the alphabet Σ, equipped with the concatenation relation $\cdot$ and predicates for the letters a. Let $<$ be the relation of strict subword (resp. stricly shorter word). Satisfaction in $\mathcal{L}$ of a bounded formula $\varphi(y_1, \ldots, y_n, S_1, \ldots, S_k)$ under an environment $[\mathbf{v}, \mathbf{R}] = [v_1, \ldots, v_n, R_1, \ldots, R_k]$ is defined via

$$\begin{aligned} &\mathcal{L} \models \exists y_{n+1} < y_i\, \varphi\, [\mathbf{v}, \mathbf{R}] < \iff \text{there is } v_{n+1} < v_i \text{ with } \mathcal{L} \models \varphi[\mathbf{v}, v_{n+1}, \mathbf{R}] \\ &\mathcal{L} \models (\mu(S_{k+1}, \ldots, S_{k+m}).(\lambda x_1.\varphi_1, \ldots, \lambda x_m.\varphi_m))(y_i)\ [\mathbf{v}, \mathbf{R}] \iff v_i \in \Gamma_1^\infty, \\ &\qquad \text{where } \Gamma = (\Gamma_1, \ldots, \Gamma_n) \text{ with } \Gamma_i(\mathbf{U}) := \{\, v \mid \mathcal{L} \models \varphi_i[v, \mathbf{R}, \mathbf{U}]\,\}. \end{aligned}$$

An m-ary relation L between words is *definable by a bounded fixed-point formula* $\varphi(y_1, \ldots, y_m)$, if $L = \{\, (u_1, \ldots, u_m) \mid \mathcal{L} \models \varphi(u_1, \ldots, u_m)\,\}$. Depending on which relation $<$ on Σ^* we use, we talk of *length-bounded* and *subword-bounded* formulas.

3.1 Bounded Fixed-Point Formulas Define Bounded Operators

The last clause in the definition of satisfaction makes sense only if we can show that the operator $\mathit{\Gamma}$ is $|\cdot|$- or $<$-bounded. We use bounded fixed-point formulas over any primitive relations and constants instead of the *CBFP*-formulas above.

Theorem 8. *Let $\mathcal{A}$ be a first-order relational structure with a well-founded quasi-ordering $\leq$ on A. Let $\varphi(x, \mathbf{y}, \mathbf{S}) := \varphi(x, y_1, \ldots, y_m, S_1, \ldots, S_n)$ be a bounded fixed-point formula such that for no set variable T, $T(x)$ is a subformula of φ. Then for all $a \in A$, $b_1 < a, \ldots, b_m < a$ and sets $R_1 \ldots R_m \subseteq A$,*

$$\mathcal{A} \models \varphi(a, b_1, \ldots, b_m, R_1, \ldots, R_n) \leftrightarrow \varphi(a, b_1, \ldots, b_m, R_1^{<a}, \ldots, R_n^{<a}). \tag{6}$$

Proof. We only consider the case $\varphi \equiv \mu(T_1, \ldots, T_k)(\lambda x_1\, \varphi_1, \ldots, \lambda x_k\, \varphi_k)(x)$: By definition, for each i we have $\mathit{freeIndV}(\varphi_i) \subseteq \{\, x_i \,\}$ and there is no subformula $Y(x_i)$ in $\varphi_i(x_i, \mathbf{S}, \mathbf{T})$ with Y among $\mathbf{S}, \mathbf{T}$. By induction, for each $a \in A$ and each sequence $\mathbf{U} = U_1, \ldots, U_k$ of subsets of A we have

$$\mathcal{A} \models \varphi_i(a, \mathbf{R}, \mathbf{U}) \leftrightarrow \varphi_i(a, \mathbf{R}^{<a}, \mathbf{U}^{<a}). \tag{7}$$

By taking $\mathbf{R}^{<a}$ instead of $\mathbf{R}$ and, respectively, $\mathbf{U}^{<a}$ instead of $\mathbf{U}$, this gives

$$\begin{aligned} &\mathcal{A} \models \varphi_i(a, \mathbf{R}^{<a}, \mathbf{U}) \leftrightarrow \varphi_i(a, \mathbf{R}^{<a}, \mathbf{U}^{<a}), \\ &\mathcal{A} \models \varphi_i(a, \mathbf{R}, \mathbf{U}^{<a}) \leftrightarrow \varphi_i(a, \mathbf{R}, \mathbf{U}). \end{aligned} \tag{8}$$

Define k-ary operators $\mathit{\Gamma}_\varphi = (\mathit{\Gamma}_{\varphi_1}, \ldots, \mathit{\Gamma}_{\varphi_k})$ and $\tilde{\mathit{\Gamma}}_\varphi = (\tilde{\mathit{\Gamma}}_{\varphi_1}, \ldots, \tilde{\mathit{\Gamma}}_{\varphi_k})$ using

$$\begin{aligned} \mathit{\Gamma}_{\varphi_i}(\mathbf{U}) &:= \{\, a \in A \mid \mathcal{A} \models \varphi_i(a, \mathbf{R}, \mathbf{U}) \,\}, \\ \tilde{\mathit{\Gamma}}_{\varphi_i}(\mathbf{U}) &:= \{\, a \in A \mid \mathcal{A} \models \varphi_i(a, \mathbf{R}^{<a}, \mathbf{U}) \,\}. \end{aligned}$$

By (8), $\mathit{\Gamma}_\varphi$ and $\mathit{\Gamma}_{\tilde{\varphi}}$ are $<$-bounded operators and by Theorem 3 have fixed points $\mathit{\Gamma}_\varphi^\infty = (\mathit{\Gamma}_{\varphi_1}^\infty, \ldots, \mathit{\Gamma}_{\varphi_k}^\infty)$ and $\tilde{\mathit{\Gamma}}_\varphi^\infty = (\tilde{\mathit{\Gamma}}_{\varphi_1}^\infty, \ldots, \tilde{\mathit{\Gamma}}_{\varphi_k}^\infty)$. Thus the formula $\varphi(x, \mathbf{S})$ has a meaning with respect to both environments $[a, \mathbf{R}]$ and $[a, \mathbf{R}^{<a}]$, given by

$$\mathcal{A} \models \varphi(a, \mathbf{R}) \iff a \in \mathit{\Gamma}_{\varphi_1}^\infty \quad \text{and} \quad \mathcal{A} \models \varphi(a, \mathbf{R}^{<a}) \iff a \in \tilde{\mathit{\Gamma}}_{\varphi_1}^\infty. \tag{9}$$

To show $\mathcal{A} \models \varphi(a, \mathbf{R}) \iff \mathcal{A} \models \varphi(a, \mathbf{R}^{<a})$, we first show that

$$\mathit{\Gamma}_\varphi^{<b} = \tilde{\mathit{\Gamma}}_\varphi^{<b} \quad \text{for all } b \leq a. \tag{10}$$

Suppose this is false. Since $<$ is well-founded, there is $b \leq a$ such that for all $c < b$, $\mathit{\Gamma}_\varphi^{<c} = \tilde{\mathit{\Gamma}}_\varphi^{<c}$, but $\mathit{\Gamma}_{\varphi_i}^{<b} \neq \tilde{\mathit{\Gamma}}_{\varphi_i}^{<b}$ for some i. But then, using (7),

$$\begin{aligned} c \in \mathit{\Gamma}_{\varphi_i}(\mathit{\Gamma}_\varphi^{<c}) &\iff c \in \mathit{\Gamma}_{\varphi_i}(\tilde{\mathit{\Gamma}}_\varphi^{<c}) \iff \mathcal{A} \models \varphi_i(c, \mathbf{R}, \tilde{\mathit{\Gamma}}_\varphi^{<c}) \\ &\iff \mathcal{A} \models \varphi_i(c, \mathbf{R}^{<c}, \tilde{\mathit{\Gamma}}_\varphi^{<c}) \iff c \in \tilde{\mathit{\Gamma}}_{\varphi_i}(\tilde{\mathit{\Gamma}}_\varphi^{<c}). \end{aligned}$$

This implies $\mathit{\Gamma}_{\varphi_i}(\mathit{\Gamma}_\varphi^{<c})^c = \tilde{\mathit{\Gamma}}_{\varphi_i}(\tilde{\mathit{\Gamma}}^{<c})^c$ for all $c < b$, which means $\mathit{\Gamma}_{\varphi_i}^{<b} = \tilde{\mathit{\Gamma}}_{\varphi_i}^{<b}$, a contradiction. Using a for c in the above calculation, the claim follows by

$$\begin{aligned} \mathcal{A} \models \varphi(a, \mathbf{R}) &\iff a \in \mathit{\Gamma}_{\varphi_1}^\infty \iff a \in \mathit{\Gamma}_{\varphi_1}(\mathit{\Gamma}_\varphi^{<a}) \\ &\iff a \in \tilde{\mathit{\Gamma}}_{\varphi_1}(\tilde{\mathit{\Gamma}}_\varphi^{<a}) \iff a \in \tilde{\mathit{\Gamma}}_{\varphi_1}^\infty \iff \mathcal{A} \models \varphi(a, \mathbf{R}^{<a}). \end{aligned}$$

Corollary 9. *On each structure $\mathcal{A}$ where $\leq$ is a well-founded quasi-ordering, every bounded fixed-point-formula $\varphi(x) := \mu(S_1, \ldots, S_n)(\lambda x_1\, \varphi_1, \ldots, \lambda x_n\, \varphi_n)(x)$ defines a simultaneous $<$-bounded operator $\Gamma_\varphi := (\Gamma_{\varphi_1}, \ldots, \Gamma_{\varphi_n})$ by*

$$\Gamma_{\varphi_i}(S_1, \ldots, S_n) := \{\, a \in A \mid \mathcal{A} \models \varphi_i(a, S_1, \ldots, S_n) \,\}.$$

In particular, the meaning of φ is well defined:

$$\mathcal{A} \models \mu(S_1, \ldots, S_n)(\lambda x_1\, \varphi_1, \ldots, \lambda x_n\, \varphi_n)(a) \iff a \in \Gamma^\infty_{\varphi_1}.$$

Fixed points Γ^∞_φ need not exist if we allow a subformula $S(x)$ in φ.[3] Positive inductive definitions ban *all* negative occurrences of recursively bound relation variables. Our class of formulas shows that this is unnecessarily restrictive in case there is a well-founded $<$ available. While we have to exclude $\mu(S)(\lambda x.\neg S(x))(x)$, we can allow formulas like $\mu(S)(\lambda x.\forall y < x.\neg S(y))(x)$.

Example 2. Let $\mathcal{A}$ be the structure $\mathcal{L}$ of words over Σ and $\leq$ be the subword-relation or the comparison by word-length.

1. Every context-free language over Σ is definable in $\mathcal{L}$ by a bounded fixed-point formula. The converse is false, since we have negation.
2. The non-context-free language $L_0 := \{\, www \mid w \in \Sigma^* \,\}$ is explicitly definable using $\mu(S)(\lambda x\, (x = \epsilon \vee \exists y < x.x = yyy))(x)$.
   The non-context-free $L_1 = \{\, a^n b^n c^n \mid n < \omega \,\}$ is definable by simultaneously defining $L_1 = L_2 L_3 \cap L_4 L_5$, with $L_2 = \{\, a^n b^n \mid n < \omega \,\}$, $L_3 = \{\, c^n \mid n < \omega \,\}$, $L_4 = \{\, a^n \mid n < \omega \,\}$, and $L_5 = \{\, b^n c^n \mid n < \omega \,\}$. This can be done as a positive induction $\mu(S_1, \ldots, S_5)(\lambda x\, \varphi_1, \ldots, \lambda x\, \varphi_5)(x)$ with, for example,
   $$\begin{aligned}\varphi_1(x, \mathbf{S}) \equiv x = \epsilon \vee [&\exists y < x\, \exists z < x\, (x = yz \wedge S_2(y) \wedge S_3(z))\\ &\wedge \exists y < x\, \exists z < x\, (x = yz \wedge S_4(y) \wedge S_5(z))].\end{aligned}$$
3. Universal quantification and boolean operations could be mixed as in the formula $\mu(S)\, (\lambda x\, \varphi)(x)$ with
   $$\varphi(x, S) := \;\; x = a \vee \forall y < x\, \forall z < x\, (x = y \cdot z \rightarrow (S(y) \leftrightarrow \neg S(z)).$$

Using Theorem 8 and Lemma 4, nested applications of bounded fixed-point operators can be combined to a single application of a simultaneous bounded fixed point operator:

Lemma 10. *('Bekič-Scott principle' for bounded recursion) Let $\varphi(x, S, T)$ and $\tilde{\varphi}(x, S, T)$ be bounded fixed-point formulas, without subformulas $U(x)$ for set variables U. Then*

$$\mathcal{L} \models \;\mu(S, T)(\lambda x.\varphi, \lambda x.\tilde{\varphi})(x) \;\leftrightarrow\; (\mu S \lambda x.\varphi[(\mu T \lambda x.\tilde{\varphi})/T])(x).$$

[3] Positive occurrences of $S(x_i)$ in φ_i could be allowed when working with locally monotone bounded operators (cf. Theorem 5).

3.2 An Invariance Property of Bounded Fixed-Point Formulas

Notions of grammaticality should have both a global and a local reading. Globally, a grammatical property $\varphi(x)$ is used to select a language

$$L(\varphi) = \{\, w \in \Sigma^* \mid \mathcal{L} \models \varphi(w) \,\}$$

from an *infinite* interpretation $\mathcal{L}$. Locally, φ should express a property of w that depends only on a *finite* substructure $\mathcal{L}(w) \subset \mathcal{L}$ and is effectively testable.

For example, note that the construction of the fixed point corresponding to a context free grammar and the construction of a recognition table for input w are related: the recognition table is a kind of 'goal oriented' selection from the stages of the inductive generation of all strings in the language.

More generally, in 'intuitively context-free' languages, grammaticality of a string w should be an 'internal' property of the string, i.e. only depend on properties of its substrings. Here the local reading of φ is its interpretation in the substructure of $\mathcal{L}$ whose universe are the subwords of w. Length-bounded fixed-point formulas $\varphi(x)$, however, can express properties of w by referring to any string v with $|v| < |w|$, and so cover some 'contextual' notions of grammaticality.

The bounded formulas all have a local reading in the sense that they are satisfied by an element w in $\mathcal{A}$ iff they are satisfied in a submodel $\mathcal{A}^{\leq w}$ defined via the quasi-ordering $\leq$ on A.

Definition 11. For a first-order relational structure $\mathcal{A}$ with a binary relation $\leq$, let $\mathcal{A}^{\leq a}$ be the substructure of $\mathcal{A}$ with universe $A^{\leq a}$ (containing the constants). A formula $\varphi(x, y_1, \ldots, y_m, S_1, \ldots, S_n)$ is $\leq$*-local in* x, if for all $\mathcal{A}$, $a \in A$, $b_1, \ldots, b_m < a$ and $S_1, \ldots, S_n \subseteq A$

$$\mathcal{A} \models \varphi(a, b_1, \ldots, b_m, S_1, \ldots, S_n) \iff \mathcal{A}^{\leq a} \models \varphi(a, b_1, \ldots, b_m, S_1^{\leq a}, \ldots, S_m^{\leq a}).$$

If $\mathcal{A}$ is $\mathcal{L}$, we write $\mathcal{L}^{\leq w}$ for the substructure of $\mathcal{L}$ whose universe consists of all subwords of w and $\mathcal{L}^{\leq |w|}$ for the substructure whose universe consists of all words of Σ^* of length at most $|w|$.

Theorem 12 (Local Substructure Invariance). *Let $\varphi(x)$ be the bounded fixed-point formula $= \mu(S_1, \ldots, S_n)(\lambda x_1\, \varphi_1, \ldots, \lambda x_n\, \varphi_n)(x)$. Then for any structure $\mathcal{A}$ with a well-founded quasi-ordering $\leq$ and any $a \in A$,*

$$\mathcal{A} \models \varphi(a) \iff \mathcal{A}^{\leq a} \models \varphi(a).$$

For $\varphi(x)$ as above, by induction on the well-founded relation $<$ the stages $\Gamma_{\varphi_i}^{\leq a}$ of the induction in $\mathcal{A}^{\leq a}$ are the intersection of those in $\mathcal{A}$ with $A^{\leq a}$. Note that *least* fixed-point formulas do not satisfy the 'local substructure invariance', because of their unbounded individual quantifiers.

As one expects, context-free grammars – as fixed-point formulas – are 'invariant under local substructures' in the sense of Theorem 12, with $\leq$ as the subword relation. Theorem 17 below shows that the converse does not hold: there are more 'intuitively context-free' than context-free languages.

3.3 Syntactic Characterization of Invariance and <-Boundedness

As is well known, a first-order formula $\varphi(x, S)$ defines a monotone operator Γ_φ iff it is logically equivalent to one where the variable S does not occur negatively. We give a similar characterization of first-order formulas $\varphi(x, S)$ that both (a) define <-bounded operators Γ_φ and (b) express 'intuitively context-free' properties. Identifying these with properties 'invariant under going to the substructure of all subwords', i.e. the 'local' properties in $\mathcal{L}$, (a) and (b) mean

$$\forall w \in \Sigma^* \, \forall S \subseteq \Sigma^* \, [\mathcal{L} \models \varphi(w, S) \iff \mathcal{L}^{\leq w} \models \varphi(w, S^{<w})].$$

In order to replace $\exists y \leq x\, \varphi$ equivalently by $\varphi[x/y] \vee \exists y < x\, \varphi$, our characterization needs $\leq$ to be antisymmetric, and hence does not cover the case of norm-bounded operators.

Definition 13. Let a first-order relational language with a binary relation $\leq$ be given. A formula $\varphi(x, y_1, \ldots, y_m, S_1, \ldots, S_n)$ is *<-bounded in x*, if for each structure $\mathcal{A}$, all $w \in A$, $v_1, \ldots, v_m < w$ and $S_1, \ldots, S_n \subseteq A$

$$\mathcal{A} \models \varphi(w, v_1, \ldots, v_m, S_1, \ldots, S_n) \iff \mathcal{A} \models \varphi(w, v_1, \ldots, v_m, S_1^{<w}, \ldots, S_n^{<w}),$$

and $\varphi(x, y_1, \ldots, y_m, S_1, \ldots, S_n)$ is (syntactically) *< x-bounded*, if all individual quantifiers are of the form $\exists y < x$ or $\forall y < x$ and each S_i occurs only in the form $S_i(y_j)$ or $S_i(y)$ for a bound variable y.

Theorem 14 (Preservation Theorem). *For a first-order formula $\varphi(x, S)$, the following conditions are equivalent:*

(i) On structures with a partial order $\leq$, $\varphi(x, S)$ is $\leq$-local and <-bounded in x.

(ii) There is a < x-bounded formula $\chi(x, S)$ such that: "$\leq$ is a partial order" $\models \varphi(x, S) \leftrightarrow \chi(x, S)$.

Our semantic proof of (i) ⇒ (ii) is too long to be included here. It derives χ as an interpolant of "$\leq$ is a partial order" $\wedge$ $\varphi^{\leq}(x, S^{<x}) \models \varphi(x, S)$. The assumption $\exists T\, (\varphi^{\leq}(x, T) \wedge T = S^{<})$ is equivalent on countable structures to a first-order theory Φ describing a consistency property for constructing T, and χ is obtained from Φ by compactness.

4 Tabular Recognizers for Bounded Fixed-Point Definitions

To evaluate monotone inductive definitions, Rounds uses an *ATM* modified by adding (i) oracle states to handle free relation variables, and (ii) recursion states that allow arbitrarily many iterations of a recursively defined predicate.

To handle bounded inductions, besides the oracle states our *ATM*'s have two new kinds of states, one for bounded individual quantification and one for bounded fixed-points.

- $\mathcal{M}_{\exists y<x.\varphi}$ has an initial $\exists$-state in which it can write to its work tape y an arbitrary string u such that $u < v$, where v is the content of the input tape x (used as a length bound resp. source for copying); control is then given to the submachine $\mathcal{M}_\varphi$ with u on its input tape y. $\mathcal{M}_{\exists y<x.\varphi}$ returns the maximum of the acceptance values of these calls to $\mathcal{M}_\varphi$.

- $\mathcal{M}_{(\mu(T_1,\ldots,T_n)(\lambda y_1.\varphi_1,\ldots))(x_1)}$ is built using the machines $\mathcal{M}_{\varphi_i(y_i,\mathbf{S},T_1,\ldots,T_n)}$. We assume oracle states for all free relation variables of the formula. Let w be the input on tape x_1 and oracles $\mathbf{R}$ for $\mathbf{S}$ be given. Let $\Gamma^{\infty}_{\varphi}(v)$ be the bit-vector of the boolean values of $v \in \Gamma^{\infty}_{\varphi_1}, \ldots, v \in \Gamma^{\infty}_{\varphi_n}$.
  First, build a 'table' $\Gamma^{<w}_{\varphi}$ of all $\Gamma^{\infty}_{\varphi}(v)$ for $v < w$: in a loop through all $v < w$, respecting $<$, check whether $\Gamma^{\infty}_{\varphi}(v)$ is already stored; if not, compute its bits $\Gamma^{\infty}_{\varphi_i}(v) = \Gamma_{\varphi_i}(\Gamma^{<v}_{\varphi})$ using the submachines $\mathcal{M}_{\varphi_i}$ with v on its input tape y_i and oracles $\mathbf{R}$ for $\mathbf{S}$ and $\Gamma^{<v}_{\varphi}$ for $\mathbf{T}$, and store the results. Second, evaluate φ_1 on input w, using $\mathcal{M}_{\varphi_1}$ with w on its input tape y_1 and oracles $\mathbf{R}$ for $\mathbf{S}$ and the 'table' $\Gamma^{<w}_{\varphi}$ for $\mathbf{T}$. Finally, return the acceptance value of $\mathcal{M}_{\varphi_1}$.

Thus we first *expand* the finite structure $\mathcal{L}^{\leq w}$ by $\Gamma^{<v}_{\varphi}$ and then can test $\mathcal{L} \models \varphi(v)$ quickly as $\mathcal{L}^{\leq w} \models \varphi_1(v, \Gamma^{<v}_{\varphi})$. In contrast to the case of least-fixed points, *before computing* $\Gamma^{<v}_{\varphi}$ *we know its size*; this could be relevant for 'bounded' queries in databases with flat quasi-ordering $\leq$. Note also that the values $\Gamma^{\infty}_{\varphi_j}(v)$ are computed only once. Whether this is an advantage over recursive computation (as used by Rounds), depends of course also on the costs of the read/write operations from/to the table.

Lemma 15. *If* $\varphi(x) := \mu(S_1, \ldots, S_n)(\lambda x_1\, \varphi_1, \ldots, \lambda x_n\, \varphi_n)(x)$ *is a bounded fixed-point formula,*

$$\{\, w \in \Sigma^* \mid \mathcal{L}^{\leq |w|} \models \varphi(w) \,\} \in \mathit{EXPTIME}, \ \{\, w \in \Sigma^* \mid \mathcal{L}^{\leq w} \models \varphi(w) \,\} \in \mathit{PTIME}.$$

Concerning the converse, note that a bounded recursive definition of the set of configurations that lead to acceptance could recur to accepting configurations of *smaller* size only, but related machine configurations have *the same* length. So it seems impossible to define all *EXPTIME* resp. *PTIME*-languages by length- resp. subword-bounded fixed-point formulas, even if we allow k-ary relation variables rather than just set variables. From a language definability point of view, this might even be expected: ordinary grammars would hardly allow multiple scanning and arbitrary rewriting(!) of an input string to test its grammaticality.

5 Tabular Recognizers for Decomposition Grammars

We now further restrict bounded inductive language definitions to obtain tabular recognition algorithms of time complexity $O(|w|^3)$ on a *RAM*. We generalize a peculiarity of *individual* quantification in the fixed-point formulation of context-free grammars which has been overlooked in Rounds' analysis. It allows to fill a field of the recognition table for input w in $O(|w|)$ steps (cf. Proposition 20 a)).

Definition 16. A bounded fixed-point definition $\mu(S_1, \ldots, S_n)(\lambda x \varphi_1, \ldots, \lambda x \varphi_n)(x)$ is *a decomposition grammar* if each $\varphi_i(x, \mathbf{S})$ is a boolean combination of formulas

$$\exists x_1 < x \ldots \exists x_k < x\, (x = t(x_1, \ldots, x_k) \wedge \psi(x_1, \ldots, x_k, S_1, \ldots, S_n)), \qquad (11)$$

where $x, x_1, \ldots, x_k$ are pairwise distinct individual variables, $t(x_1, \ldots, x_k)$ is a term (here: a word made of x_i's and constants a) in which each variable occurs at most once, and $\psi(\mathbf{x}, \mathbf{S})$ is a conjunction of formulas $S_i(x_j)$ and their negations. The grammar is *in normal form*, if in each subformula (11) either $k = 0$ and $t \equiv a$ for some $a \in \Sigma$, or $k = 2$ and $t \equiv x_1 \cdot x_2$.

We can test property (11) by 'decomposing' its argument string x into strict substrings $x_1, \ldots, x_k$ according to the pattern t and checking which of the predicates $S_1, \ldots, S_k$ hold true of the substrings $x_1, \ldots, x_k$.

Each language $L \subseteq \Sigma^*$ definable by decomposition grammars can also be defined by decompostion grammars in normal form. We could allow ψ in (11) to be a boolean combination of $S_j(x_i)$'s; having disjuncts there amounts to delaying decisions in the parsing process. We could also allow μ-formulas in ψ and have nested recursive definitions. This is possible since the class of definable languages is closed under substitution, for which we need Lemma 10.

Theorem 17. *The class of languages definable by decomposition grammars contains the context-free languages, is closed under the boolean operations* $\cup, \cap, \neg$, *the regular operations of* $\cdot, ^*$, *and under substitution.*

Decomposition grammars seem equally expressive as the *hierarchical complement intersection grammars* of Heilbunner and Schmitz [5], but we have not checked the details.

Before turning to the recognition complexity of decomposition grammars, we generalize these to allow a *relational* interpretation of the syntactic categories.

5.1 Head Grammars and Languages of Segmented Strings

In the syntax of natural languages, concatenation is not the only primitive used to combine expressions. Other operations have been studied (cf. [10, 11]), such as the insertion of one string into another one, or the wrapping of a splitted string around the 'head' (for example: stem) of another one. Technically, one uses *string pairs* over Σ and the following *wrapping operations* $\circ_i : \Sigma^{*2} \times \Sigma^{*2} \to \Sigma^{*2}$

$$(v_1, v_2) \circ_1 (w_1, w_2) := (v_1 w_1, w_2 v_2) \qquad (v_1, v_2) \circ_2 (w_1, w_2) := (v_1, w_1 w_2 v_2)$$

$$(v_1, v_2) \circ_3 (w_1, w_2) := (v_1 w_1 w_2, v_2) \qquad (v_1, v_2) \circ_4 (w_1, w_2) := (v_1, v_2 w_1 w_2)$$

$$(v_1, v_2) \circ_5 (w_1, w_2) := (v_1 v_2 w_1, w_2)$$

More generally, we consider *segmented strings* $(w_1, \ldots, w_m)$, i.e. strings $w = w_1 \cdots w_m$ segmented into several consecutive substrings $w_1, \ldots, w_m$. These are useful at various places in language description: on the word level to decompose a string w into segments such as verb stem, prefixes, infixes and suffixes, or on the phrasal level to handle 'discontinuous constituents', such as a noun phrase whose noun and relative clause are separated by the verb. For simplicity, we fix the number m of segments; but we enlarge the class of operations:

Definition 18. Let *Op* be the set of operations $\circ : (\Sigma^*)^m \times (\Sigma^*)^m \to (\Sigma^*)^m$ that are definable by

$$(x_1, \ldots, x_m) \circ (y_1, \ldots, y_m) := (v_1, \ldots, v_m), \tag{12}$$

where $v_1, \ldots, v_m$ are words over $\{ x_1, \ldots, x_m, y_1, \ldots, y_m \}$ and each x_i and y_j occurs exactly once[4] in $v_1 \cdots v_m$. Let *the structure of m-fold segmented strings* be

$$\mathcal{L}_m := ((\Sigma^*)^m, \circ, (a, \epsilon, \ldots, \epsilon), \ldots, (\epsilon, \ldots, \epsilon, a))_{a \in \Sigma, \circ \in Op}.$$

Definition 19. A *normal form decomposition grammar for m-fold segmented strings* is a formula $\varphi(x) = \mu(S_1, \ldots, S_n)(\lambda x \varphi_1, \ldots, \lambda x \varphi_n)(x)$, where each formula $\varphi_i(x, S_1, \ldots, S_n)$ is a boolean combination of formulas

$$\exists x_1 < x \, \exists x_2 < x \, (x = t(x_1, x_2) \wedge \psi(x_1, x_2, S_1, \ldots, S_n)), \tag{13}$$

in which either $t \equiv a$ for some $a \in \Sigma$, or $t \equiv (x_1 \circ x_2)$ for some $\circ \in Op$, and ψ is a conjunction of formulas $S_i(x_j)$ and their negations. *The language of m-fold segmented strings defined by* $\varphi(x)$ is

$$L(\varphi) := \{ (w_1, \ldots, w_m) \mid \mathcal{L}_m \models \varphi((w_1, \ldots, w_m)) \}.$$

To interprete $<$, use comparison by the length $|(w_1, \ldots, w_m)| = |w_1| + \cdots + |w_m|$ of segmented strings. A *head grammar* is a normal form decomposition grammar for 2-fold segmented strings over the set $Op = \{ \circ_1, \ldots, \circ_5 \}$, where each φ_i is a disjunction of formulas (13) in which ψ has no negations.

C. Pollard[10] introduced head grammars, in a more complicated 'rewriting' format, as context-free grammars where concatenation is replaced by the (non-associative) wrapping operations $\circ_1, \ldots, \circ_5$. Formal properties of head languages have been studied by K. Roach[11].

5.2 Complexity of Recognition with Decomposition Grammars

Let $\mu(S_1, \ldots, S_n)(\lambda x_1 \, \varphi_1, \ldots, \lambda x_n \, \varphi_n)(x)$ be a decomposition grammar, and $w \in \Sigma^*$. In order to decide whether $\mathcal{L} \models \mu(S_1, \ldots, S_n)(\lambda x_1 \, \varphi_1, \ldots, \lambda x_n \, \varphi_n)(w)$, one proceeds as follows:

1. For each subword v of w and each $i \leq n$, compute the boolean value $\Gamma^{\infty}_{\varphi_i}(v)$, and store the bitvector $(\Gamma^{\infty}_{\varphi_1}(v), \ldots, \Gamma^{\infty}_{\varphi_n}(v))$ as field $M(v)$ of a table with $|subwords(w)|$ many fields.
2. To compute $\Gamma^{\infty}_{\varphi_i}(v)$, where $\varphi_i(x, \mathbf{S})$ is a boolean combination of subformulas φ as in (iii), determine the corresponding values $\Gamma^{\infty}_{\varphi}(v)$ as explained in (iii) and evaluate the boolean combination of the results.
3. To compute $\Gamma^{\infty}_{\varphi}(v)$ for $\varphi(x, \mathbf{S}) = \exists x_1 < x \ldots \exists x_k < x \, (x = t(\mathbf{x}) \wedge \psi(\mathbf{x}, \mathbf{S}))$, where $\psi(\mathbf{x}, \mathbf{S})$ is a boolean combination of formulas $S_i(x_j)$ with $1 \leq i \leq n$ and $1 \leq j \leq k$,
   (a) determine all splittings of v into substrings $v_1, \ldots, v_k$ such that $v = t(v_1, \ldots, v_k)$, and
   (b) for each such splitting $\mathbf{v}$, evaluate $\psi(\mathbf{v}, \mathbf{S})$ by looking up the values for $S_i(v_j)$ in $M(v_j)$.

[4] After finishing this work, I found the same restriction used in the 'multiple context-free grammars' of Seki e.a.[13] for a complexity result related to Theorem 21.

The complexity of computing the table M for an input w is proportional to the number of substrings v of w times the cost of computing a field $M(v)$. Since only splittings into strict substrings are allowed in (iii), the table M can be filled by computing $M(v)$ with subwords v of w of increasing length.

Remark. If $\Gamma = \Gamma_\varphi$ for a context-free grammar $\varphi(x)$ in Chomsky normal form and $<$ is the strict subword relation, $\Gamma(\Gamma^{<w})$ is the familiar recognition table for w in the algorithm of Cocke-Younger-Kasami. Different ways of computing $\Gamma(\Gamma^{<w})$ correspond to different versions of the recognizer. In an off-line version, one can compute $\Gamma(\Gamma^{<v})$ for all subwords v of *increasing length*. On-line versions compute $\Gamma(\Gamma^{<v})$ for *increasing prefixes* v of w: if $w = va$ ends in a letter a we first determine the table $\Gamma(\Gamma^{<v})$ of the prefix v, then that of the next input symbol, $\Gamma(\Gamma^{<a})$, and finally compute $\Gamma(\Gamma^{<ua})$ for increasing suffixes u of v.

We now estimate the number of RAM-steps needed to construct a recognition table $M(w_1, \ldots, w_m)$ for a decomposition grammar for m-segmented strings.

Proposition 20. *Let $\circ$ be any such $2m$-ary operation as just defined.*

a) There are $O(\max_i |w_i|^m)$ many decompositions $(u_1, \ldots, u_m) \circ (v_1, \ldots, v_m)$ of $(w_1, \ldots, w_m) \in (\Sigma^*)^m$.
b) The table $M(w_1, \ldots, w_m)$ has $O(\max_i |w_i|^{2m})$ fields.

Proof. a) Each of the v_i that define $\circ$ contains at least one of the variables $x_1, \ldots, y_m$. To consume the remaining m variables one needs m applications of concatenations (giving adjacent variables in the $v_1, \ldots, v_m$). To find substrings of $w_1, \ldots, w_m$ that match the variables, we therefore have to find m splitting positions $i_0, \ldots, i_m$ in $w_1, \ldots, w_m$. There are $O(|w|^k)$ many k-tuples $i_1 \leq i_2 \ldots \leq i_k \leq |w|$ in a string w and so $O(|w_1|^{k_1} \cdots |w_m|^{k_m})$ many splittings with k_i splitting positions in w_i. It follows that there are at most $O(\max |w_i|^m)$ splittings $(u_1, \ldots, u_m) \circ (v_1, \ldots, v_m) = (w_1, \ldots, w_m)$.

b) For a subword $(v_1, \ldots, v_m) \leq (w_1, \ldots, w_m)$ of $(w_1, \ldots, w_m)$, the v_i must be empty or occur as non-overlapping subwords of the $w_1, \ldots, w_m$. The total number of k nonoverlapping subwords of w_i is bounded by the beginning and end positions $i_1 \leq j_1 \leq \ldots \leq i_k \leq j_k \leq |w_i|$ of the subwords, which makes $O(|w_i|^{2k})$ possibilities. There are $O(|w_1|^{2k_1} \cdots |w_m|^{2k_m})$ subwords $(u_1, \ldots, u_m) \leq (w_1, \ldots, w_m)$ such that k_i out of the $u_1, \ldots, u_m$ are nonoverlapping subwords of w_i. Hence the number of all subwords $(u_1, \ldots, u_m) \leq (w_1, \ldots, w_m)$ is bounded by the sum of values $O(|w_1|^{2k_1} \cdots |w_m|^{2k_m})$ over all $k_1, \ldots, k_m$ where $\sum k_i = m$, giving $O(\max_i |w_i|^{2m})$. Multiplying a) and b), we get

Theorem 21. *For any normal form decomposition grammar $\varphi(x)$ for m-segmented strings, a recognition table M for input $(w_1, \ldots, w_m)$ can be constructed in $O(\max_i |w_i|^{3m})$ many RAM-steps.*

For $m = 1$, this gives the familiar $O(|w|^3)$ bound for language recognition with respect to context-free grammars in Chomsky normal form. For $m = 2$, we get an $O(|(w_1, w_2)|^6)$ bound for head grammars in 'Chomsky' normal form.

6 Open Problems

Bounded fixed-point formulas may be applied on other structures, like the finite trees with the subtree relation. They might also be useful to develop relational query languages with low complexities, since the number and size of the inductive stages depend on the quasi-ordering rather than the size of the domain.

An extension of our monadic bounded inductive definability to the n-ary case should present no difficulties. We also expect that the (subword-) boundedly definable languages strictly contain the boolean closure of context-free languages.

We have indicated why the inclusion of subword- resp. length-bounded inductive definability in *PTIME* resp. *EXPTIME* should be strict. A precise characterization of the boundedly definable languages in terms of complexity is open.

Acknowledgement: This work has partially been supported by ESPRIT BRA 7230, GENTZEN.

References

1. A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133, 1981.
2. H.-D. Ebbinghaus and J. Flum. *Finite Model Theory.* Springer, Berlin 1995.
3. Y. Gurevich and S. Shelah. Fixed-point extensions of first-order logic. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science*, 1985.
4. M. Harrison. *Introduction to Formal Languages.* Addison Wesley, Reading 1978.
5. S. Heilbrunner and L. Schmitz. An efficient recognizer for the boolean closure of context-free languages. *Theoretical Computer Science*, 80:53–75, 1991.
6. G. Hotz and G. Pitsch. Fast uniform analysis of coupled-context-free languages. In S. Abiteboul and E. Shamir, editors, *21st International Colloquium on Automata, Languages and Programming*, pages 412–423. Lecture Notes in Computer Science 820, Springer, Berlin 1994.
7. N. Immerman. Relational queries computable in polynomial time. *Information and Control*, 68:86–104, 1986.
8. C. Lautemann, T. Schwentick, and D. Thérien. Logics for context-free languages. In *Computer Science Logic '94*, pages 205–216. Lecture Notes in Computer Science 933, Springer, Berlin 1995.
9. H. Leiß. Towards Kleene Algebra with Recursion. In E. Börger e.a., editors, *Computer Science Logic '91*, pages 242–256. Lecture Notes in Computer Science 626, Springer, Berlin 1992.
10. C. Pollard. *Generalized Phrase Structure Grammars, Head Grammars, and Natural Language.* PhD thesis, Department of Linguistics, Stanford University, 1984.
11. K. Roach. Formal properties of head grammars. In A. Manaster-Ramer, editor, *Mathematics of Language*, pages 293–348. John Benjamins, Amsterdam 1987.
12. W. Rounds. A logic for linguistic descriptions and an analysis of its complexity. *Computational Linguistics*, 14(4):1–9, 1988.
13. H. Seki, T. Matsumura, M. Fujii and T. Kasami. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229, 1991.
14. M. Vardi. Complexity of relational query languages. In *14th ACM Symposium on the Theory of Computing*, pages 137–146, 1982.
15. K. Vijay-Shanker and A. Joshi. Some computational properties of tree adjoining grammars. In *Proceedigs of the 23rd Meeting of the Association for Computational Linguistics*, pages 82–93, Chicago 1988.