Solving Fixed-Point Equations by Derivation Tree Analysis[⋆]

Javier Esparza and Michael Luttenberger

Institut für Informatik, Technische Universität München, 85748 Garching, Germany
{esparza,luttenbe}@in.tum.de

Abstract. Systems of equations over ω-continuous semirings can be mapped to context-free grammars in a natural way. We show how an analysis of the derivation trees of the grammar yields new algorithms for approximating and even computing exactly the least solution of the system.

1 Introduction

We are interested in computing (or approximating) solutions of *systems of fixed-point equations* of the form

$$\begin{aligned} X_1 &= f_1(X_1, X_2, \ldots, X_n) \\ X_2 &= f_2(X_1, X_2, \ldots, X_n) \\ &\vdots \\ X_n &= f_n(X_1, X_2, \ldots, X_n) \end{aligned}$$

where $X_1, X_2, \ldots, X_n$ are variables and $f_1, f_2, \ldots, f_n$ are n-ary functions over some common domain S. Fixed-point equations are a natural way of describing the equilibrium states of systems with n interacting components (particles, populations, program points, etc.). Loosely speaking, the function f_i describes the next state of the i-th component as a function of the current states of all components, and so the solutions of the system describe the equilibrium states. In computer science, a prominent example of fixed-point equations are dataflow equations. In this case, the system is a program, the components are the control points of the program, the common domain is some universe of data facts, and the f_i's describe the dataflow to (or from) the i-th control point to all other control points in one program step (see e.g. [NNH99]).

Without further assumptions on the functions $f_1, \ldots, f_n$ and the domain S, little can be said about the existence and computability of a solution. In the last years we have studied *polynomial systems* (systems in which the f_i's are multivariate polynomials) in which S is an ω*-continuous semiring*, a well-known algebraic structure [Kui97]. This setting has the advantage that the system always has a least solution, a result usually known as Kleene's theorem [Kui97], which allows us to concentrate on the task of approximating or computing it.

This paper surveys recent results [EKL07a,EKL07b,EKL08a,EKL10,Lut10] and some work in progress [Lut]. The presentation emphasizes the connection between the

[⋆] This work was partially supported by the project "Polynomial Systems on Semirings: Foundations, Algorithms, Applications" of the Deutsche Forschungsgemeinschaft.

algebraic study of equations and formal language theory. In fact, our main goal is to show how equations can be mapped to context-free grammars in a natural way,[1] and how an analysis of the derivation trees of the grammars yields new algorithms for approximating and even computing the least solution of the equations.

The paper is structured as follows. After some preliminaries (Section 2), we introduce a known result (Section 3): the least solution of a system is equal to the *value* of its associated grammar, where the value of a grammar is defined as the sum of the values of its derivation trees, and the value of a derivation tree is defined as the (ordered) product of its leaves. This connection allows us to approximate the least solution of a system by computing the values of "approximations" to the grammar. Loosely speaking, a grammar G_1 approximates G_2 if every derivation tree of G_1 is a derivation tree of G_2 up to irrelevant details. We show that Kleene's theorem, which not only proves the existence of the least solution, but also provides an algorithm for approximating it, corresponds to approximating G by grammars $G^{[1]}, G^{[1]}, \ldots$ where $G^{[h]}$ generates the derivation trees of G of height h. We then introduce (Section 4) a faster approximation by grammars $H^{[1]}, H^{[1]}, \ldots$ where $H^{[h]}$ generates the derivation trees of G of *dimension* h [EKL08a,EKL10]. We show that this approximation is a generalization of Newton's method for approximating the zero of a differentiable function, and present a new result about its convergence speed when multiplication is commutative [Lut][2]. In the final part of the paper (Section 5) we apply the insights obtained from Newton's and Kleene's approximation to different classes of *idempotent* semirings, i.e., semirings in which the law $a + a = a$ holds. We obtain approximation algorithms that actually provide the exact solution after a finite number of steps.

2 Polynomial Equations Over Semirings

For the definition of polynomial systems we need a set S and two binary operations on S, *addition* and *multiplication*, satisfying the usual associativity and distributivity laws:

Definition 1. *A* semiring *is a tuple* $\langle S, +, \cdot, 0, 1\rangle$*, where (i)* S *is a set with* $0, 1 \in S$ *called the carrier of the semiring, (ii)* $\langle S, +, 0\rangle$ *is a commutative monoid with neutral element* 0*, (iii)* $\langle S, \cdot, 1\rangle$ *is a monoid with neutral element* 1*, (iv)* 0 *is an annihilator, i.e.* $0 \cdot a = a \cdot 0 = 0$ *for all* $a \in S$*, and (v) multiplication distributes over addition from the left and from the right.*

When addition and multiplication are clear from the context, we identify a semiring $\langle S, +, \cdot, 0, 1\rangle$ with its carrier S . We also often write ab for $a \cdot b$. A polynomial over a semiring S is a finite sum of finite products of variables and semiring elements. For instance, if X, Y denote variables and $a, b, c \in S$ denote semiring elements, then $aYb + XYXc$ is a polynomial. Notice that multiplication is not required to be commutative, and so we cannot represent a single polynomial in *monomial form*, i.e. as a finite sum of products of the form $aX_1 \cdots X_m$, where $a \in S$ is a coefficient and $X_1 \cdots X_n$ is

[1] We do not claim to be the first to come up with this connection. See e.g. [BR82,Boz99].
[2] The proof has not yet been published, but we feel confident it is correct.

a product of variables. Things change for polynomial *systems*. In this case, we may introduce auxiliary variables following the procedure used to put a context-free grammar in Chomsky normal form; for instance, the univariate equation

$$X = aXb + XcX + e$$

which is not in monomial form, can be transformed into the multivariate system

$$X = aXY + XZ + e \qquad Y = b \qquad Z = cX$$

which *simulates* the original system w.r.t. the X-component. Although our results do not require systems to be in monomial form, for this survey we always assume it to simplify notation.

Polynomial systems over semirings may have no solution. For instance, $X = X + 1$ has no solution over the reals. However, if we extend the reals with a maximal element ∞ (correspondingly adapting addition and multiplication so that these operations still are monotone), we can consider ∞ a solution of this equation. We restrict ourselves to semirings with these "limit" elements.

Definition 2. *Given a semiring S, define the binary relation $\sqsubseteq$ by*

$$a \sqsubseteq b :\Leftrightarrow \exists d \in S : a + d = b.$$

*A semiring S is ω-*continuous *if (i) $\langle S, \sqsubseteq \rangle$ is a ω-complete partial order, i.e., the supremum $\sup_{i\in\mathbb{N}} a_i$ of any ω-chain $a_0 \sqsubseteq a_1 \sqsubseteq \ldots$ exists in S w.r.t. the partial order $\sqsubseteq$ on S; and (ii) both addition and multiplication are ω-continuous in both arguments, i.e., for any ω-chain $(a_i)_{i\in\mathbb{N}}$ and semiring element a:*

$$a + \sup_{i\in\mathbb{N}} a_i = \sup_{i\in\mathbb{N}}(a + a_i) \qquad \textit{and} \qquad a \cdot \sup_{i\in\mathbb{N}} a_i = \sup_{i\in\mathbb{N}}(a \cdot a_i)$$

and symmetrically in the other argument.

We adopt the following convention:

If not stated otherwise, S denotes an ω-continuous semiring $\langle S, +, \cdot, 0, 1\rangle$.

In an ω-continuous semiring we can extend the summation operator $\sum$ from finite to countable families $(a_i)_{i\in I}$ by defining

$$\sum_{i\in I} a_i := \sup \left\{ \sum_{i \in F} a_i \mid F \subseteq I, |F| < \infty \right\}.$$

It then can be shown that $\sum$ is still associative and multiplication distributes over $\sum$ from both the left and the right [DKV09]. Note that in a ω-continuous semiring S we have $0 \sqsubseteq a$ for all $a \in S$. Hence, the reals extended by ∞ do not constitute an ω-continuous semiring w.r.t. the canonical order $\leq$, but the *nonnegative* reals do.

It is easy to see that *Kleene's fixed-point theorem* applies to polynomial systems over ω-continuous semirings:[3]

[3] The theorem is often also attributed to Tarski. In fact, it can be seen as a slight extension of Tarski's fixed-point theorem for complete lattices [Tar55], or as a particular case of Kleene's first recursion theorem [Kle38].

Theorem 1 ([Kui97]). *Every polynomial system $X = f(X)$ over an ω-continuous semiring has a least solution μf w.r.t. $\sqsubseteq$, and μf is equal to the supremum of the* Kleene sequence*:*

$$0 \sqsubseteq f(0) \sqsubseteq f(f(0)) \sqsubseteq \ldots \sqsubseteq f^i(0) \sqsubseteq f^{i+1}(0) \sqsubseteq \ldots \tag{1}$$

Observe that Kleene's theorem not only guarantees the existence of the least fixed point, but also provides a first approximation method, usually called fixed-point iteration.

3 From Equations to Grammars

We illustrate by means of examples how Kleene's theorem allows us to connect polynomial systems of equations with context-free grammars and the derivation trees associated with them. For a formal presentation see e.g. [Boz99,EKL10,DKV09].

Consider the equation

$$X = \frac{1}{4}X^2 + \frac{1}{4}X + \frac{1}{2} \tag{2}$$

over the nonnegative reals extended by ∞, which is an ω-continuous semiring. The equation is equivalent to $(X-1)(X-2) = 0$, and so its least solution is $X = 1$. We introduce identifiers a, b, c for the coefficients, yielding the formal equation

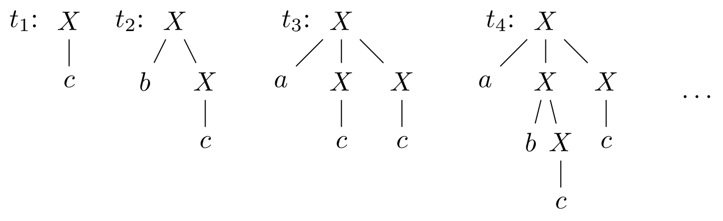
$$X = f(X) := aX^2 + bX + c\,. \tag{3}$$

We say that (2) is an *instance* of (3). Formally, instances correspond to valuations. A *valuation* is a mapping $V\colon \Sigma \to S$, where Σ is the set of identifiers of the formal equation (in our example $\Sigma = \{a, b, c\}$), and S is an ω-continuous semiring. So (2) is the instance of (3) for the valuation where S are the nonnegative reals with ∞, $V(a) = V(b) = 1/4$, and $V(c) = 1/2$. We denote the instance for V by $X = f_V(X)$, and its least solution by μf_V.

We associate a context-free grammar with Equation (3) by reading every summand of the right-hand side as a production:

$$G:\ X \to aXX \mid bX \mid c\,, \tag{4}$$

We denote by $T(G)$ the set of derivation trees of G. We depict derivation trees in the standard way as ordered finite trees and say that a derivation tree $t \in T(G)$ *yields* a word $a_1a_2\ldots a_l \in \Sigma^*$ if the i-th leaf from the left of t is labeled by a_i. For instance, the following trees t_1, t_2, t_3, t_4 yield the words $c, bc, acc, abcc$, respectively:



Note that the grammar G for an univariate equation has only a single nonterminal, and thus the axiom of G is clear. In the case of a multivariate polynomial system $X = f(X)$, we construct in the same way a context-free grammar G, but without an explicit axiom. $T(G)$ stands for the union of the sets $T_1(G), \ldots, T_n(G)$ of derivation trees corresponding to setting $X_1, \ldots, X_n$ as axiom. In the following, we do not explicitly distinguish between the univariate and the multivariate case, and adopt the convention:

> Given a grammar G without explicit axiom, a result regarding G or $T(G)$ is to be understood as holding for any possible choice of the axiom.

A valuation $V \colon \Sigma \to S$ extends naturally to the derivation trees of G: for a tree $t \in T(G)$ yielding $a_1 a_2 \ldots a_l$, we define

$$V(t) = V(a_1) \cdot V(a_2) \cdot \ldots \cdot V(a_l),$$

and for a set of trees $T \subseteq T(G)$, we define $V(T) = \sum_{t \in T} V(t)$. For instance, for the trees t_1, t_2, t_3, t_4 shown in the picture above and the valuation mapping a, b, c to $1/4$, $1/4$, and $1/2$, respectively, we get

$$V(\{t_1, t_2, t_3, t_4\}) = V(t_1)+V(t_2)+V(t_3)+V(t_4) = 1/2+1/8+1/16+1/64 = 45/64.$$

Now, as a last step, we can extend V to a valuation of the complete grammar.

Definition 3. *Let G be the grammar of a formal polynomial system $X = f(X)$, and let $V \colon \Sigma \to S$ be a valuation over some ω-continuous semiring S. We define $V(G) = V(T(G)) = \sum_{t \in T(G)} V(t)$ over S.*

The starting point of our paper is a well-known result stating that, given a formal polynomial equation $X = f(X)$ and a valuation V, the least solution of $X = f_V(X)$ is equal to $V(G)$ (see e.g. [Boz99], and, independently, [EKL10]; the essence of the result can be traced back to [BR82,Tha67,CS63]). In other words, the least solution can be obtained by adding the values under V of all its derivation trees.

Theorem 2 ([Boz99,EKL10]). *Let $X = f(X)$ be a formal polynomial system with a set Σ of formal identifiers, and let $V \colon \Sigma \to S$ be a valuation. Then:*

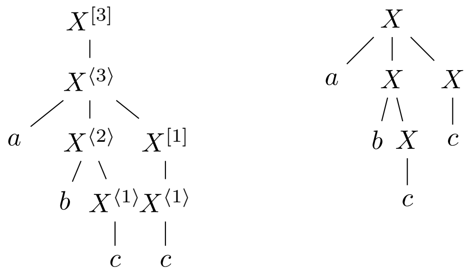
$$\mu f_V = V(G). \tag{5}$$

By our convention, for a multivariate system Theorem 5 states that for every variable X_i the X_i-component of μf_V is given by the infinite sum of all evaluated derivation trees derivable from X_i w.r.t. G.

We sketch a proof of this theorem for the particular case of equation (3). Let us "unfold" the grammar G of (4) by augmenting the nonterminal X with a counter keeping track of the height of a derivation:

$$\begin{aligned}
X^{\langle 1\rangle} &\to c\\
X^{[1]} &\to X^{\langle 1\rangle}\\
X^{\langle 2\rangle} &\to aX^{\langle 1\rangle}X^{\langle 1\rangle} \mid bX^{\langle 1\rangle}\\
X^{[2]} &\to X^{\langle 2\rangle} \mid X^{[1]}\\
X^{\langle 3\rangle} &\to aX^{\langle 2\rangle}X^{\langle 2\rangle} \mid aX^{[1]}X^{\langle 2\rangle} \mid aX^{\langle 2\rangle}X^{[1]} \mid bX^{\langle 2\rangle}\\
X^{[3]} &\to X^{\langle 3\rangle} \mid X^{[2]}\\
&\vdots\\
X^{\langle h\rangle} &\to aX^{\langle h-1\rangle}X^{\langle h-1\rangle} \mid aX^{[h-2]}X^{\langle h-1\rangle} \mid aX^{\langle h-1\rangle}X^{[h-2]} \mid bX^{\langle h-1\rangle}\\
X^{[h]} &\to X^{\langle h\rangle} \mid X^{[h-1]}\\
&\vdots
\end{aligned}$$

Let $G^{[h]}$ ($G^{\langle h\rangle}$) be the grammar consisting of those "unfolded" rules whose left-hand side is given by one of the variables of $\mathcal{X}^{[h]} = \{X^{\langle 0\rangle}, X^{[0]}, \ldots, X^{\langle h\rangle}, X^{[h]}\}$, taking $X^{[h]}$ ($X^{\langle h\rangle}$) as axiom.[4] An easy induction shows the existence of a bijection between $T(G^{[h]})$ ($T(G^{\langle h\rangle})$) and the trees of $T(G)$ of height at most (exactly) h. In fact, it is easy to see that $G^{[h]}$ ($G^{\langle h\rangle}$) and G are both unambiguous[5], and the bijection just assigns to a tree of $T(G^{[h]})$ the unique tree of G yielding the same word. For instance, the tree of $G^{[3]}$ shown on the left of the figure below is mapped to the tree of G of height 3 shown on the right:



Hence, $V(G^{[h]})$ ($V(G^{\langle h\rangle})$) is the contribution to $V(G)$ of the derivation trees of height at most (exactly) h to $V(G)$. It therefore suffices to show that $f_V^h(0) = V(G^{[h]})$. Note that by the extension of V to derivation trees, $V(G^{[h]})$ and $V(G^{\langle h\rangle})$ can be computed recursively as follows (with $a_V := V(a), b_V := V(b), c_V := V(c)$):

$$\begin{aligned}
V(G^{\langle h\rangle}) &= a_V V(G^{\langle h-1\rangle})^2 + a_V V(G^{[h-2]})V(G^{\langle h-1\rangle})\\
&\quad + a_V V(G^{\langle h-1\rangle})V(G^{[h-2]}) + b_V V(G^{\langle h-1\rangle})\\
V(G^{[h]}) &= V(G^{[h-1]}) + V(G^{\langle h\rangle})
\end{aligned}$$

[4] In the multivariate case, for every choice Z of the axiom of G, define $G^{[h]}$ ($G^{\langle h\rangle}$) analogously with $Z^{[h]}$ ($Z^{\langle h\rangle}$) as axiom.

[5] A grammar G is *unambiguous* if for every word $w \in L(G)$ there is a unique derivation (tree) w.r.t. G.

where $V(G^{\langle 1 \rangle}) = c_V$ and $V(G^{[-1]}) := V(G^{[0]}) := 0$.

Now, an easy induction proves the stronger claim that

$$f_V^h(0) = V(G^{[h]}) \text{ and } f_V^h(0) = f_V^{h-1}(0) + V(G^{\langle h \rangle})$$

and by Kleene's theorem we get $\mu f_V = \sup_{h \in \mathbb{N}} f_V^h(0) = \sup_{h \in \mathbb{N}} V(G^{[h]}) = V(G)$.

Notice that this proof not only reduces the problem of computing the least solution of $X = f_V(X)$ to the problem of computing $V(G)$, it also shows that:

Kleene's approximation sequence is the result of evaluating the derivation trees of G by increasing height.

4 Newton's Approximation

In the last section we have constructed grammars $G^{\langle 1 \rangle}, G^{\langle 2 \rangle}, \ldots$ that, loosely speaking, "partition" the derivation trees of G according to height. Formally, there is a bijection between the derivation trees of $G^{\langle h \rangle}$ and the derivation trees of G of height exactly h. Using these grammars we can construct grammars $G^{[1]}, G^{[2]}, \ldots$ such there is a bijection between the derivation trees of $G^{[h]}$ and the derivation trees of G of height at most h. The grammars $G^{[h]}$ allow us to iteratively compute approximations $V(G^{[h]})$ to $V(G) = \mu f_V$.

We can transform this idea into a general principle for developing approximation algorithms. Given a grammar G, we say that a sequence $(G^{\langle i \rangle})_{i \in \mathbb{N}}$ of grammars *partitions* G if $T(G_i) \cap T(G_j) = \emptyset$ for $i \neq j$, and there is a bijection between $\bigcup_{i \in \mathbb{N}} T(G^{\langle i \rangle})$ and $T(G)$ that preserves the yield, i.e., the yield of a tree is equal to the yield of its image under the bijection.. Every sequence $(G^{\langle i \rangle})_{i \in \mathbb{N}}$ that partitions G induces another sequence $(G^{[i]})_{i \in \mathbb{N}}$, defined as in the previous section, such that $T(G^{[i]}) = \bigcup_{j \leq i} T(G^{\langle i \rangle})$. We say that $(G^{[i]})_{i \in \mathbb{N}}$ *converges* to G. The following proposition follows easily from these definitions.

Proposition 1. *Let $X = f(X)$ be a formal polynomial system with a set Σ of formal identifiers, and let G be the context-free grammar associated to it. If a sequence $(G_i)_{i \in \mathbb{N}}$ of grammars converges to G, then*

$$\mu f_V = \sup_{i \in \mathbb{N}} V(G_i) \; .$$

The unfolding of the last section assigns to *every* variable in the right-hand-side of a production a lower index (height) than the variable on the left-hand-side, which forbids any kind of unbounded recursion in the unfolded grammars. We now unfold the grammar G so that *nested-linear recursion* is allowed [EKL08b,GMM10]. Again we augment each variable X by a counter, yielding variables $X^{\langle i \rangle}, X^{[i]}$. A derivation starting from $X^{\langle i \rangle}$ ($X^{[i]}$) allows for exactly i (at most i) nested-linear recursions. For the grammar (4) we get:

$$\begin{array}{ll}
X^{\langle 1\rangle} & \to c \mid bX^{\langle 1\rangle} \\
X^{[1]} & \to X^{\langle 1\rangle} \\
X^{\langle 2\rangle} & \to aX^{\langle 1\rangle}X^{\langle 1\rangle} \mid aX^{[1]}X^{\langle 2\rangle} \mid aX^{\langle 2\rangle}X^{[1]} \mid bX^{\langle 2\rangle} \\
X^{[2]} & \to X^{\langle 2\rangle} \mid X^{[1]} \\
& \vdots \\
X^{\langle i\rangle} & \to aX^{\langle i-1\rangle}X^{\langle i-1\rangle} \mid aX^{[i-1]}X^{\langle i\rangle} \mid aX^{\langle i\rangle}X^{[i-1]} \mid bX^{\langle i\rangle} \\
X^{[i]} & \to X^{\langle i\rangle} \mid X^{[i-1]} \\
& \vdots
\end{array}$$

It is instructive to compare the productions of $X^{\langle h\rangle}$ in Kleene's approximation, and the productions of $X^{\langle i\rangle}$ as defined above:

$$\begin{array}{lllll}
X^{\langle h\rangle} \to aX^{\langle h-1\rangle}X^{\langle h-1\rangle} & \mid aX^{[h-2]}X^{\langle h-1\rangle} & \mid aX^{\langle h-1\rangle}X^{[h-2]} & \mid bX^{\langle h-1\rangle} \\
X^{\langle i\rangle} \to aX^{\langle i-1\rangle}X^{\langle i-1\rangle} & \mid aX^{[i-1]}X^{\langle i\rangle} & \mid aX^{\langle i\rangle}X^{[i-1]} & \mid bX^{\langle i\rangle}
\end{array}$$

Let $H^{[i]}$ ($H^{\langle i\rangle}$) denote the grammar with axiom $X^{[i]}$ ($X^{\langle i\rangle}$) and consisting of those productions "reachable" from $X^{[i]}$ ($X^{\langle i\rangle}$) in the above unfolding. As in the case of Kleene approximation, we can easily show by induction that $H^{[i]}$ is unambiguous, and that the mapping assigning to a tree of $T(H^{[i]})$ the unique tree of G deriving the same word is a bijection. Since every word of $L(G)$ belongs to $L(H^{[i]})$ for some $i \in \mathbb{N}$, the sequence $(H^{[i]})_{i\in\mathbb{N}}$ converges to G.

Again, we can compute $V(H^{\langle i\rangle})$ and $V(H^{[i]})$ recursively where $\mu X.g(X)$ denotes the the least solution of the equation $X = g(X)$ (again $a_V := V(a), \ldots$):

$$\begin{array}{ll}
V(H^{\langle i\rangle}) & := \mu X.(\, a_V X V(H^{[i-1]}) + a_V V(H^{[i-1]})X + b_V X + a_V V(H^{\langle i-1\rangle})^2 \,) \\
V(H^{[i]}) & := V(H^{\langle i\rangle}) + V(H^{\langle i-1\rangle})
\end{array}$$

where $V(H^{\langle 1\rangle}) := \mu X.(\, b_V X + c_V \,)$.

At this point the reader may ask whether any progress has been made: instead of solving the polynomial system $X = f_V(X)$ we have to solve the polynomial systems $X = g_i(X)$. However, these systems are linear, while $X = f_V(X)$ may be nonlinear, and in ω-continuous semirings solving linear equations reduces to computing the Kleene star $a^* := \sum_{i\in\mathbb{N}} a^i$. So, for any ω-continuous semiring which allows for an efficient computation of a^*, this approximation scheme becomes viable. For instance, over the nonnegative reals we have $a^* = \frac{1}{1-a}$ if $a < 1$ and $a^* = \infty$ otherwise. Thus, if V is a valuation on the real semiring, then the solution of a linear equation can be easily computed. For the equations above elementary arithmetic yields

$$V(G^{\langle i\rangle}) := \frac{a_V V(G^{\langle i-1\rangle})^2}{1 - 2a_V V(G^{[i-1]}) - b_V} \qquad V(G^{[i]}) := V(G^{\langle i\rangle}) + V(G^{[i-1]}) \tag{6}$$

with $V(G^{[1]}) = V(G^{\langle 1\rangle}) := \dfrac{c_V}{1 - b_V}$.

The following table compares the first approximations obtained by using the approximation schemes derived in this and the previos section for our example (2):

Kleene	$V(G^{\langle i \rangle})$	$1/2$	$3/8$	$105/1024$	$\dots$
	$V(G^{[i]})$	$1/2$	$11/16$	$809/1024$	$\dots$
Newton	$V(H^{\langle i \rangle})$	$2/3$	$4/15$	$16/255$	$\dots$
	$V(H^{[i]})$	$2/3$	$14/15$	$254/255$	$\dots$

(7)

It is now time to explain why we call this scheme Newton's approximation. For every valuation V over the reals, the least solution of $X = f_V(X)$ is a zero of the polynomial $g(X) = f_V(X) - X = a_V X^2 + (b_V - 1)X + c_V$. Again, an easy induction shows:

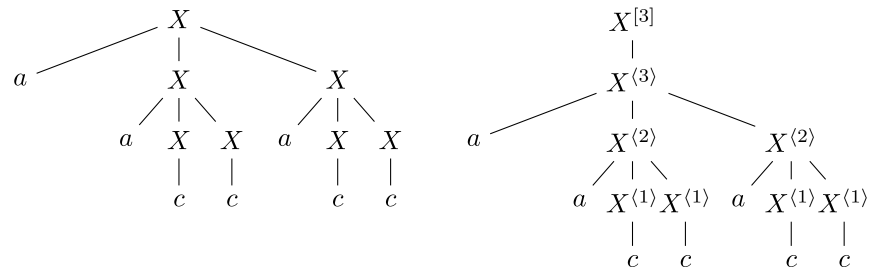
$$V(H^{\langle i \rangle}) = -\frac{g(V(H^{[i-1]}))}{g'(V(H^{[i-1]}))} \qquad V(H^{[i]}) = -\frac{g(V(H^{[i-1]}))}{g'(V(H^{[i-1]}))} + V(H^{[i-1]})$$

starting now from $V(H^{\langle 0 \rangle}) = V(H^{[0]}) = 0$, where $g'(X)$ denotes the derivative of g – in our example: $g'(X) = 2a_V X + b_V - 1$. These equations are nothing but Newton's classical method for approximating the solution of $g(X) = 0$ starting at the point 0, and this is not a coincidence: we have recently shown that this relation holds for every polynomial equation $X = f_V(X)$ over the nonnegative reals [EKL10]. So this approximation scheme generalizes Newton's method to equations over arbitrary ω-continuous semirings.

Recall that Kleene's approximation corresponds to evaluating the derivation trees of G by increasing height. The question whether we can charaterize Newton's approximation in a similar way has been answered positively in [EKL10]. We need the notion of dimension of a derivation tree.

Definition 4. *Let t be a derivation tree. If t consists of a single node, then its dimension is 1. Otherwise, let d be the maximal dimension of the children of t. If two or more children have dimension d, then t has dimension $d+1$; otherwise, t has dimension d.*

For instance, the derivation tree of the grammar (4) shown below on the left has dimension 3 (its second and third child have dimension 2, because both of them have two children of dimension 1).



We can prove:

Theorem 3 ([EKL10]). *For every $i \geq 1$, there is a yield-preserving bijection between $T(H^{[i]})$ and the trees of $T(G)$ of dimension at most i.*

According to this theorem, the tree above must belong to $T(H^{[3]})$ and indeed this is the case, as shown by the derivation tree on the right. Note that along any path from the root to a leaf the sequence of numbers in the superscripts drops atmost by one in each step. One the other hand, moving from a leaf to the root, the superscript only increases from i to $i+1$ at a given node if this very node has at least a second child with superscript i. The superscripts in round (square) brackets happen just to be (an upper bound on) the dimension of the corresponding subtree. So we conclude:

$$\boxed{\text{Newton's approximation sequence is the result of evaluating the derivation trees of } G \text{ by increasing dimension.}}$$

4.1 Convergence of Newton's method in commutative semirings

The convergence speed of Newton's method over the reals is well-understood. In many cases – for example (7) – it converges *quadratically*, which in computer science terms means that the approximation error decreases exponentially in the number of iterations. In this section we analyze the convergence speed valid for *arbitrary* commutative semirings, i.e., semirings in which multiplication is commutative.

Recall that, by definition,

$$V(H^{[i]}) = \sum_{t \in T(H^{[i]})} V(t) \qquad \text{and} \qquad V(G) = \sum_{t \in T(G)} V(t) \ .$$

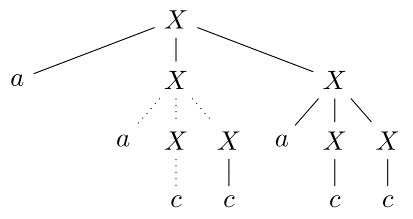
For every $s \in S$, let $\alpha^{[i]}(s)$ be the number of trees $t \in T(H^{[i]})$ such that $V(t) = s$, if the number is finite, and $\alpha^{[i]}(s) = \infty$ otherwise. Define $\alpha(s)$ similarly for $T(G)$. Then we have

$$V(H^{[i]}) = \sum_{s \in S} \sum_{i=1}^{\alpha^{[i]}(s)} s \qquad V(G) = \sum_{s \in S} \sum_{i=1}^{\alpha(s)} s$$

with the convention $\sum_{i=1}^{0} s = 0$. We estimate the convergence speed of Newton's method by analyzing how fast the sequence $(\alpha^{[i]}(s))_{i \in \mathbb{N}}$ converges to $\alpha(s)$. Our result shows that in a system of n equations after $(kn+1)$ iterations of Newton's method we have $\alpha^{[kn+1]}(s) \geq \min\{\alpha(s), k\}$.

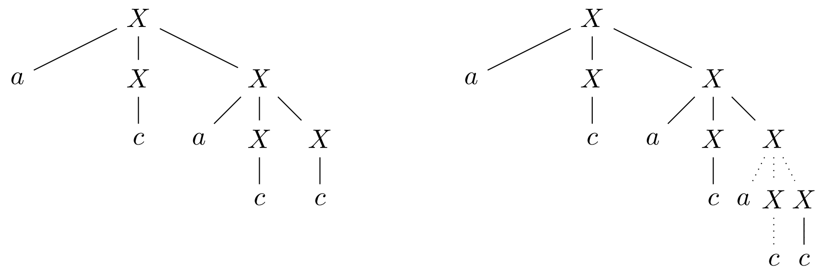
Theorem 4 ([Lut]). *Let $X = f(X)$ be a formal polynomial system with n equations, and let V be a valuation over a commutative ω-continuous semiring S. We have $\alpha^{[k \cdot n+1]}(s) \geq \min\{\alpha(s), k\}$ for every $s \in S$ and every $k \in \mathbb{N}$.*

We sketch the proof of the theorem for the (very) special case $n = k = 1$. We have to show $\alpha^{[2]}(s) \geq \min\{\alpha(s), 1\}$, i.e., that $\alpha(s) > 0$ implies $\alpha^{[2]}(s) > 0$ or, equivalently, that for every $t \in T(G)$ some $t' \in T(H^{[2]})$ satisfies $V(t) = V(t')$. As $T(H^{[2]})$ is in bijection with the trees of $T(G)$ of dimension at most 2, it suffices to prove that for every $t \in T(G)$ there is $t' \in T(G)$ of dimension at most 2 such that $V(t') = V(t)$. If t has dimension 1 or 2, we take $t' = t$. Otherwise, we explain how to proceed using grammar (4) and the tree of dimension 3 deriving the word $aaccacc$:



If we remove the dotted subtree (pump tree), the dimension of the second child of the root decreases by 1, and we are left with the tree of dimension 2 shown below, on the left. This tree only derives the word $acacc$, and so the idea is to reinsert the missing subtree so that the result (i) is again a derivation tree w.r.t. G, and (ii) we do not increase the dimension. If we achieve this, then the new tree derives a permutation w of $acacacc$ and, *since the semiring is commutative*, we have $V(w) = V(aaccacc)$. Condition (i) poses no problem in the univariate case, as as all inner nodes correspond to the same variable (nonterminal). In order to satisfy condition (ii), it suffices to pick any subtree derived from X of dimension 2 and replace the edge to its father by the missing dotted subtree as shown below, on the right.

It can be shown that this reallocation of subtrees is also possible in the multivariate case and allows to generate the required number of distinct derivations trees, although additional care is needed in order to satisfy the two conditions.



5 Derivation tree analysis for idempotent semirings

In the previous section, we have seen how to relocate subtrees of a derivation tree in order to reduce its dimension. In commutative semirings, relocating subtrees preserves the value of the tree, and we have used this fact to derive Theorem 4, a quantitative meassure of the speed at which the Newton approximations $V(H^{[i]})$ converge to $V(G)$. In particular, for $k = 1$ we obtain $\alpha^{[n+1]}(s) \geq \min\{\alpha(s), 1\}$ or, equivalently,

For every tree $t \in T(G)$ there is a tree $t' \in V(H^{[i]})$ such that $V(t) = V(t')$.

This has an important consequence for *idempotent* semirings, i.e., for semirings satisfying the identity $a + a = a$ for every $a \in S$. For any valuation V over an idempotent semiring, $V(t) = V(t')$ implies $V(t) + V(t') = V(t')$. So for idempotent and commutative ω-continuous semirings we get $V(G) + V(H^{[n+1]}) = V(H^{[n+1]})$, which together with $V(H^{[n+1]}) \sqsubseteq V(G)$ implies $V(H^{[n+1]}) = V(G)$. It follows:

Theorem 5 ([EKL10]). *Let $X = f(X)$ be a formal polynomial system with n equations. For every valuation V over an idempotent and commutative ω-continuous semiring*

$$\mu f_V = V(H^{[n+1]}) \ .$$

Intuitively, this result states that in order to compute μf_V we can safely "forget" the derivation trees of dimension greater than $n+1$, which implies that Newton's method terminates after at most $n+1$ iterations.

In the rest of the section we study two further classes of idempotent ω-continuous semirings for which a similar result can be proved: idempotence allows to "forget" derivation trees, and compute the least solution exactly after finitely many steps.

5.1 1-bounded semirings

A semiring $\langle S, +, \cdot, 0, 1\rangle$ 1-*bounded* if it is idempotent and $a \sqsubseteq 1$ for all $a \in S$.

One-bounded semirings occur, for instance, in probabilistic settings when one is interested in the most likely path between two nodes of a Markov chain. The probability of a path is the product of the probabilities of its transitions, and we are interested on the maximum over all paths. This results in an equation system over the *Viterbi semiring* [DKV09] whose carrier is the interval $[0, 1]$, and has $\max$ and $\cdot$ as addition and multiplication operators, respectively.

We show that over 1-bounded semirings we may "forget" all derivation trees of height greater than n. Fix a formal polynomial system $X = f(X)$ with n equations and valuation V over a 1-bounded semiring. Let G be the associated context-free grammar. G then has also n nonterminals. A derivation tree $t \in T(G)$ is *pumpable* if it contains a path from its root to one of its leaves in which some variable occurs at least twice. Clearly, every tree of height at least $n+1$ is pumpable. It is well-known that a pumpable tree t induces a *pumpable* factorization $w = uvxyz$ of its yield w such that $uv^ixy^iz \in L(G)$ for every $i \geq 0$. In particular, for every $i \geq 0$ there is a derivation tree t^i that (i) yields uv^ixy^iz, and (ii) is derived from the same axiom as t. Now we have

$$\begin{aligned} V(t) + V(t^0) &= V(w) + V(uxz) \\ &= V(u)V(v)V(x)V(y)V(z) + V(u)V(x)V(z) \\ &\sqsubseteq V(u)\ 1\ V(x)\ 1\ V(z) + V(u)V(x)V(z) && \text{(1-boundedness)} \\ &= V(uxz) && \text{(idempotence)} \\ &= V(t^0) \end{aligned}$$

Repeating this procedere as long as possible, we eventually arrive from a pumpable tree t to another tree $\hat{t}$ of height at most n with $V(t) + V(\hat{t}) = V(\hat{t})$. So, denoting by $T^{[n]}(G)$ the trees of G of height at most n, we have

Theorem 6 ([EKL08a]). *For $X = f(X)$ a formal polynomial system in n variables, G its associated grammar, and V any valuation over a 1-bounded semiring, we have:*

$$\mu f_V = V(T(G)) = V(T^{[n]}(G)) = V(G^{[n]}) = f_V^h(0).$$

Since the Kleene sequence converges after at most n steps we can compute the least solution even if the semiring is not ω-continuous.

5.2 Star-distributive semirings

In an ω-continuous semiring we can define the *Kleene star* operation by $a^* = \sum_{i \geq 0} a^i$, where $a^0 = 1$. A semiring $\langle S, +, \cdot, 0, 1 \rangle$ is *star-distributive* if it is ω-continuous, idempotent, commutative, and $(a + b)^* = a^* + b^*$ holds for every $a, b \in S$.

The *tropical semiring* $\langle \overline{\mathbb{N}}, \min, +, \infty, 0 \rangle$ is a prominent example of star-distributive semiring. Actually, any ω-continuous commutative and idempotent semiring in which the natural order $\sqsubseteq$ is total is star-distributive. Indeed, for any two elements a, b, assuming w.l.o.g. $a \sqsubseteq b$, which implies $a^* \sqsubseteq b^*$, we get:

$$(a + b)^* = a^* = a^* + b^* \ .$$

Finally, for a last bit of motivation, a recent paper shows that the computation of several types of provenance of datalog queries can be reduced to the problem of (in our terminology) computing the least solution of a formal polynomial system over a commutative semiring S [GKT07]. Specifically, in the case of the why-provenance S is also idempotent and further augmented by the identity $a^2 = a$ for all $a \in \Sigma$. Clearly, such semirings are star-distributive.

We show that idempotent together with commutativity and star-distributivity allows us to forget most derivation trees of a grammar G associated with a formal polynomial system. In fact, we do not use star-distributivity directly, but the following two identities implied by it in conjunction with commutativity:

Proposition 2. *If S is star-distributive, then for every $a, b \in S$*

$$a^* + b^* = a^* b^* \qquad \textit{and} \qquad (ab^*)^* = a^* + ab^*.$$

Again, fix a formal polynomial system $X = f(X)$ with n equations, and let G be the grammar (without explicit axiom) associated to the system. Further, let V be a valuation over some star-distributive semiring S. We have:

Proposition 3. *Let $t \in T(G)$ be a pumpable tree deriving a word w with pumpable factorization $uvxyz$. Then there are pumpable trees $t_1, \ldots, t_r \in T(G)$ (derived from the same axiom as t) of height at most $n + 1$ such that each t_i has a pumpable factorization $u_i v_i x_i y_i z_i$ satisfying*

$$V(w) \sqsubseteq \sum_{i=1}^{r} \sum_{j=0}^{\infty} V(u_i v_i^j x_i y_i^j z_i) \ . \tag{8}$$

In essence, this proposition tells us that we only need to evaluate derivation trees of G which are either of "unpumpable" (thus, of height at most n) or the result of pumping a fixed factorization in a pumpable derivation tree of height at most $n + 1$, while we may "forget" the rest.

We sketch one case of the proof of the proposition. Fix a pumpable tree t with pumpable factorization $uvxyz$ as schematically described in Figure 1(a) where the middle (grey) and the lower (dark grey) part are derived from the same nonterminal, and the top part (white) may be empty. If t has height at most $n + 1$, we set $t_1 := t$ and are

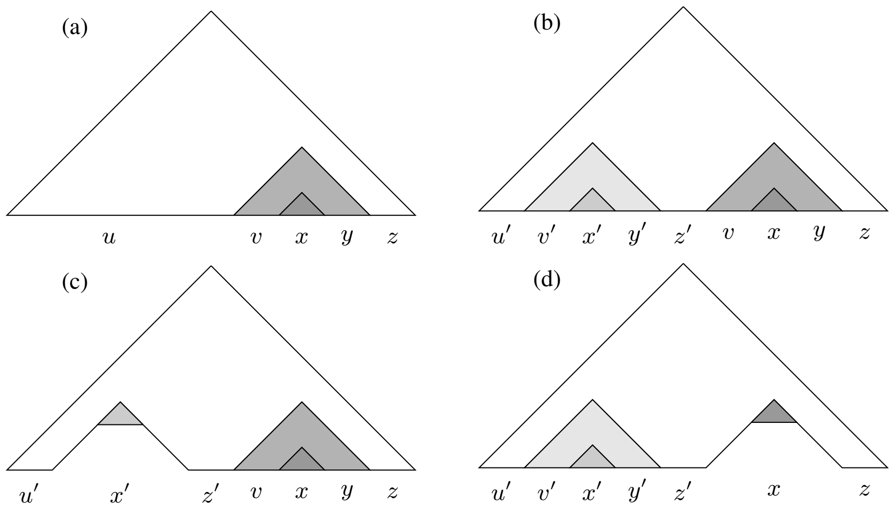


Fig. 1. "Unpumping" trees.

done. Otherwise, one of the three parts of t (white, grey, or dark grey) contains a subtree of height at least $n+1$. Since G only has n variables, this subtree is also pumpable. We only consider the case that the pumpable tree is on the left part of the white zone (other cases are similar). Then there is a pumpable factorization of u, i.e. $u = u'v'x'y'z'$, as shown in Figure 1(b), and we have $u'(v')^i x'(y)^i z' u v^j x y^j z \in L(G)$ for every $i, j \geq 0$. Applying the properties of star-distributive semirings we get

$$\begin{aligned}
&\sum_{i\geq 0}\sum_{j\geq 0} u'(v')^i x'(y')^i z' v^j x y^j z \\
&= u'x'z'xz(v'y')^*(vy)^* && \text{(commutativity)} \\
&= u'x'z'xz((v'y')^* + (vy)^*) && (a^*b^* = a^* + b^*) \\
&= \sum_{i\geq 0} u'(v')^i x'(y')^i z' xz + \sum_{j\geq 0} u'x'z'v^j x y^j z
\end{aligned}$$

It is easy to see that G has derivation trees t_1 and t_2 (schematically shown in Figure 1(c) and (d)) with pumpable factorizations $w_1 = u_1v_1x_1y_1z_1$ and $w_2 = u_2v_2x_2y_2z_2$ given by

$$\begin{array}{lllll}
u_1 = u'x'z' & v_1 = v & x_1 = x & y_1 = y & z_1 = z \\
u_2 = u' & v_2 = v' & x_2 = x' & y_2 = y' & z_2 = z'xz
\end{array}$$

Therefore, we have

$$V(w) \sqsubseteq \sum_{j=0}^{\infty} V(u_1 v_1^j x_1 y_1^j z_1) + \sum_{j=0}^{\infty} V(u_2 v_2^j x_2 y_2^j z_2)$$

If t_1 and t_2 have height at most $n+1$, then we are done; otherwise, the step above is iterated. This concludes the proof sketch.

Let us now see how to apply the proposition. Let $L \subseteq L(G)$ be the language containing

- the words derived by the "unpumpable" trees of G, and
- the words of the form uv^jxy^jz, where $uvxyz$ is a pumpable factorization of a tree of $T(G)$ of height at most $n+1$.

Given $w \in L(G)$, there are two possible cases: if w is derived by some "umpumpable" tree, then $w \in L$, and so $V(w) \sqsubseteq V(L)$; if w is derived by some pumpable tree, then by (8) we also have $V(w) \sqsubseteq V(L)$. So $V(w) \sqsubseteq V(L)$ holds for every $w \in L(G)$. Since S is idempotent, we get

$$\begin{aligned} V(G) &= \textstyle\sum_{t\in T(G)} V(t) & \\ &= \textstyle\sum_{w\in L(G)} V(w) & \text{(idempotence)} \\ &\sqsubseteq \textstyle\sum_{w\in L(G)} V(L) & (V(w) \sqsubseteq V(L)) \\ &= V(L) & \text{(idempotence and } \omega\text{-continuity)} \end{aligned}$$

Looking at the definition of L it is not difficult to show (see [EKL08a]) that it is subsumed by the words of $L(G)$ derived by the *bamboos* of $T(G)$, a set of derivation trees defined as follows:

Definition 5. *A derivation tree t is a* bamboo *if there is a path leading from the root of t to some leaf of t, the* stem, *such that the height of every subtree of t not containing a node of the stem is at most n.*

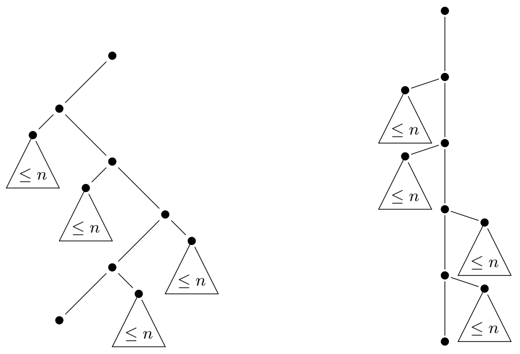


Fig. 2. An example of the structure of a bamboo: it consists of a stem of unbounded length from which subtrees of height at most n sprout; on the right it is shown with its stem straightened.

Figure 2 illustrates the definition. The definition of "bamboo" directly leads to an unfolding rule for G: in every rule we limit the recursion depth of all but one terminal to n in the same way as we did in the case of the Kleene approximation. Notice that, since $V(G) = V(L)$ by idempotence, we do not need to ensure that each derivation tree of the unfolded grammars uniquely corresponds to a derivation tree of G. This very much simplifies the definition of the unfolding. For instance, if G has nonterminals $\{X, Y, Z, U, V\}$, then the productions

$$X \to aXY \mid bZ \mid c$$

are unfolded to

$$\begin{array}{rcl} X & \to & aXY^{[5]} \mid aX^{[5]}Y \mid bZ \mid c \\ X^{[5]} & \to & aX^{[4]}Y^{[4]} \mid bZ^{[4]} \mid c \\ & \dots & \\ X^{[2]} & \to & aX^{[1]}Y^{[1]} \mid bZ^{[1]} \mid c \\ X^{[1]} & \to & c \end{array}$$

The structure of the grammar then again allows us to recursively compute the yield of the derivations trees derived from any nonterminal which gives us an algorithm for computing the least fixed point of any formal polynomial system w.r.t. any valuation over some star-distributive semiring:

Theorem 7 ([EKL08a]). *Let $X = f(X)$ be formal polynomial system consisting of n equations and let V be a valuation over a star-distributive semiring S.*

Then μf_V can be computed using n Kleene iteration steps and then solving a single linear system over S.

This result can be used to compute the provenance of datalog queries over the tropical semiring, a problem that was left open in [GKT07].

6 Conclusions

We have presented some old and some new links between computational algebra and language theory. We have shown how the formal similarity between fixed-point equations and context-free grammars goes very far, and leads to novel algorithms.

The unfolding of grammars leading to Newton's approximation has already found some applications in verification [GMM10,EG11] and Petri net theory [GA11]. Theorem 5 has lead to a simple algorithm for constructing an automaton whose language is Parikh-equivalent to the language of a given context-free grammar [EGKL11]. Theorem 7 was used in [EKL08a] to improve the complexity bound of [CCFR07] for computing the throughput of context-free grammars from $\mathcal{O}(n^4)$ to $\mathcal{O}(n^3)$.

An interesting question is whether the results we have obtained can be proved by purely algebraic means, e.g. without using "tree surgery". Further open questions concern data structures and efficient algorithms for the approximation schemes we have sketched.

Acknowledgments

Many thanks to Volker Diekert for his help with Theorem 4, to Rupak Majumdar for pointing us to applications of semirings to the provenance problem in databases [GKT07], and to Pierre Ganty for many discussions.

References

[Boz99] S. Bozapalidis. Equational elements in additive algebras. *Theory Comput. Syst.*, 32(1):1–33, 1999.

[BR82] J. Berstel and C. Reutenauer. Recognizable formal power series on trees. *Theor. Comput. Sci.*, 18:115–148, 1982.

[CCFR07] D. Caucal, J. Czyzowicz, W. Fraczak, and W. Rytter. Efficient computation of throughput values of context-free languages. In *CIAA'07*, LNCS 4783, pages 203–213. Springer, 2007.

[CS63] N. Chomsky and M.P. Schützenberger. *Computer Programming and Formal Systems*, chapter The Algebraic Theory of Context-Free Languages, pages 118 – 161. North Holland, 1963.

[DKV09] M. Droste, W. Kuich, and H. Vogler. *Handbook of Weighted Automata*. Springer, 2009.

[EG11] J. Esparza and P. Ganty. Complexity of pattern-based verification for multithreaded programs. In *POPL*, pages 499–510, 2011.

[EGKL11] J. Esparza, P. Ganty, S. Kiefer, and M. Luttenberger. Parikhs theorem: A simple and direct automaton construction. *Inf. Process. Lett.*, 111(12):614–619, 2011.

[EKL07a] J. Esparza, S. Kiefer, and M. Luttenberger. An extension of newton's method to ω-continuous semirings. In *DLT*, pages 157–168, 2007.

[EKL07b] J. Esparza, S. Kiefer, and M. Luttenberger. On fixed point equations over commutative semirings. In *STACS*, pages 296–307, 2007.

[EKL08a] J. Esparza, S. Kiefer, and M. Luttenberger. Derivation tree analysis for accelerated fixed-point computation. In *DLT*, pages 301–313, 2008.

[EKL08b] J. Esparza, S. Kiefer, and M. Luttenberger. Newton's method for ω-continuous semirings. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 14–26. Springer, 2008.

[EKL10] J. Esparza, S. Kiefer, and M. Luttenberger. Newtonian program analysis. *J. ACM*, 57(6):33, 2010.

[GA11] P. Ganty and M. Atig. Approximating Petri net reachability along context-free traces. Technical report, arXiv:1105.1657v1, 2011.

[GKT07] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, pages 31–40, 2007.

[GMM10] P. Ganty, R. Majumdar, and B. Monmege. Bounded underapproximations. In *CAV*, pages 600–614, 2010.

[Kle38] S. C. Kleene. On notation for ordinal numbers. *J. Symb. Log.*, 3(4):150–155, 1938.

[Kui97] W. Kuich. *Handbook of Formal Languages*, volume 1, chapter 9: Semirings and Formal Power Series: Their Relevance to Formal Languages and Automata, pages 609 – 677. Springer, 1997.

[Lut] M. Luttenberger. An extension of Parikh's theorem. Technical report, Technische Universität München, Institut für Informatik. Forthcoming.

[Lut10] M. Luttenberger. *Solving Systems of Polynomial Equations: A Generalization of Newton's Method*. PhD thesis, Technische Universität München, 2010.

[NNH99] F. Nielson, H.R. Nielson, and C. Hankin. *Principles of Program Analysis*. Springer, 1999.

[Tar55] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific J. Math.*, 5(2):285–309, 1955.

[Tha67] J. W. Thatcher. Characterizing derivation trees of context-free grammars through a generalization of finite automata theory. *J. Comput. Syst. Sci.*, 1(4):317–322, 1967.