Solving fixed-point equations on semirings

Javier Esparza
Technical University of Munich

Joint work with
Stefan Kiefer, Michael Luttenberger, Maximilian Schlund
Fixed-point equations

We study systems of equations of the form

\[
\begin{align*}
X_1 &= f_1(X_1, \ldots, X_n) \\
X_2 &= f_2(X_1, \ldots, X_n) \\
& \quad \vdots \\
X_n &= f_n(X_1, \ldots, X_n)
\end{align*}
\]

where the \( f_i \)'s are polynomials over an \( \omega \)-continuous semiring.
\( \omega \)-continuous semirings

\( \omega \)-continuity:

the relation \( a \sqsubseteq b \iff \exists c : a + c = b \) is a partial order

\( \sqsubseteq \)-chains have limits

Examples: nonnegative integers and reals with \( \infty \), tropical semiring, min-max semirings, complete lattices, Viterbi and Łukasiewicz semirings, language semiring . . .

In the rest of the talk:

\text{semiring} \equiv \omega \text{-continuous semiring.}
Research program

Develop **generic** solution or approximation methods, valid for all semirings or at least large classes.
Kleene iteration

**Theorem [Kleene]:** A system of fixed-point equations over a semiring has a least solution $\mu f$ w.r.t. the natural order $\sqsubseteq$.

This least solution is the supremum of $\{k_i\}_{i \geq 0}$, where

\[
\begin{align*}
k_0 &= f(0) \\
k_{i+1} &= f(k_i)
\end{align*}
\]

**Basic algorithm for calculation of $\mu f$:** Compute $k_0, k_1, k_2, \ldots$ until either $k_i = k_{i+1}$ or the approximation is considered adequate.
The left-linear case

\[
\begin{align*}
X_1 &= a_{11}X_1 + \cdots + a_{1n}X_n + b_1 \\
X_2 &= a_{21}X_1 + \cdots + a_{2n}X_n + b_2 \\
&\vdots \\
X_n &= a_{n1}X_1 + \cdots + a_{nn}X_n + b_n
\end{align*}
\]
The left-linear case

(Loosely speaking!) Kleene iteration has linear convergence over the reals: $k$ iterations give $\Theta(k)$ accurate bits.
(Loosely speaking!) Kleene iteration has linear convergence over the reals: $k$ iterations give $\Theta(k)$ accurate bits.

**Gauss elimination:**

Define $a^* := 1 + a + a \cdot a + a \cdot a \cdot a + \cdots$

**Arden’s Lemma:** the least solution of $X = aX + b$ is $X := a^* b$

**Algorithm:** pick $X_i$ and rewrite its equation as $X_i = aX_i + b$; replace $X_i$ in all other equations by $a^* b$
The left-linear case

Gauss elimination reduces solving a system of left-linear equations to computing $a^*$ for a given semiring element $a$. 

(What does it mean to “solve” an equation?)
The left-linear case

Gauss elimination reduces solving a system of left-linear equations to computing \( a^* \) for a given semiring element \( a \).

Real semiring: either \( a^* = 0, \ a^* = 1/(1 - a), \) or \( a^* = \infty \)
Gauss elimination reduces solving a system of left-linear equations to computing $a^*$ for a given semiring element $a$.

Real semiring: either $a^* = 0$, $a^* = 1/(1 - a)$, or $a^* = \infty$

Language semiring: we use $a^*$ as representation of $\sum_{i=0}^{\infty} a^i$

(What does it mean to “solve” an equation?)
The non-linear case

Real semiring: convergence of Kleene iteration can be very slow.

\[ X = \frac{1}{2} X^2 + \frac{1}{2} \quad \mu f = 1 = 0.99999\ldots \]

Logarithmic convergence: \( k \) iterations give \( \Theta(\log k) \) accurate bits.

For example, \( k_{2000} = 0.9990 \)
The non-linear case

Real semiring: convergence of Kleene iteration can be very slow.

\[ X = \frac{1}{2} X^2 + \frac{1}{2} \]

\[ \mu f = 1 = 0.99999\ldots \]

Logarithmic convergence: \( k \) iterations give \( \Theta(\log k) \) accurate bits.

For example, \( k_{2000} = 0.9990 \)

No reduction to computing Kleene stars
Kleene Iteration for $X = f(X)$ (univariate case)
Kleene Iteration for $X = f(X)$ (univariate case)
Kleene Iteration for $X = f(X)$ (univariate case)
Kleene Iteration for $X = f(X)$ (univariate case)
Kleene Iteration for $X = f(X)$ (univariate case)
Kleene Iteration for $X = f(X)$ (univariate case)
Kleene Iteration for $X = f(X)$ (univariate case)
Kleene Iteration for $X = f(X)$ (univariate case)
Newton’s Method for \( X = f(X) \) (univariate case)
Newton’s Method for $X = f(X)$ (univariate case)
Newton’s Method for $X = f(X)$ (univariate case)
Newton’s Method for $X = f(X)$ (univariate case)
Newton's Method for $X = f(X)$ (univariate case)
Newton’s Method for $X = f(X)$ (univariate case)
Newton’s method is usually very efficient

Often \textit{exponential} convergence*: \( k \) iterations give \( \Theta(2^k) \) bits.

but not robust.

May not converge, converge only locally (in some neighborhood of the least fixed-point), or converge very slowly.

* Called quadratic convergence in numerical mathematics.
A frustrating mismatch

- Kleene Iteration is robust and applicable to every semiring, but converges slowly.
- Newton’s Method may converge very fast, but is not robust and can only be applied to the reals.
A frustrating mismatch and its solution

- Kleene Iteration is robust and applicable to every semiring, but converges slowly.

- Newton’s Method may converge very fast, but is not robust and can only be applied to the reals.

Main results:

- Newton’s Method can be generalized to arbitrary semirings, and becomes as robust as Kleene’s method (our work).

- Newton’s method converges at least linearly and often exponentially over the real semiring (some work by us + work by Etessami, Stewart and Yannakakis).
Generalizing Newton’s Method
Derivation trees I

An equation $X = f(X)$ over a semiring induces a context-free grammar $G$

Examples:

\[
X = 0.3 X^2 + 0.5 \quad \text{induces} \quad X \rightarrow 0.3 X X \mid 0.5
\]

\[
X = 0.2 XY + 0.3 \quad \text{induces} \quad X \rightarrow 0.2 X Y \mid 0.8
\]

\[
Y = 0.7 XY + 0.1 \quad \text{induces} \quad Y \rightarrow 0.7 X Y \mid 0.1
\]

Running example with arbitrary semiring elements $a, b, c$:

\[
X = a X^2 + b X + c \quad \text{and} \quad G : X \rightarrow a X X \mid b X \mid c
\]
Derivation trees II

Assign to a derivation tree \( t \) its yield

\[ Y(t) : = (\text{ordered}) \text{ product of the leaves of } t \]

Assign to a set \( T \) of derivation trees its yield

\[ Y(T) : = \text{sum of the yields of the elements of } T \]

\[ G : X \rightarrow aXX \mid bX \mid c \]

Yield of the three trees: \( c + a \cdot c \cdot c + a \cdot c \cdot b \cdot c \)
Proposition: Let $D$ be the set of all derivation trees of $G$. Then

$$\mu f = Y(D)$$
Proposition: The $i$-th Kleene approximant $k_i$ is the yield of all derivation trees of height at most $i$. 

\[ X = f(X) \quad \text{Trees of depth} \leq i \quad G \]

\[ k_i \]
Approximants as yields: Newton iteration

**Theorem:** The $i$-th Newton approximant $\nu_i$ is the yield of all derivation trees of Strahler number at most $i$.

\[
X = f(X) \quad G \\
\downarrow \quad \downarrow \\
\nu_i \quad \text{Trees of Strahler number } \leq i
\]
HYPSOMETRIC (AREA-ALTITUDE) ANALYSIS OF EROSIONAL TOPOGRAPHY

By Arthur N. Strahler
Which is the main stream?
The “finger-tip” channels constitute the first-order segments. […]
A second-order segment is formed by the junction of any two first-order streams; a third-order segment is formed by the joining of any two second order streams, etc.
Streams of lower order joining a higher order stream do not change the order of the higher stream
Strahler number of a tree

Definition: Strahler number $S(t)$ of a tree $t$:

If $t$ has no subtrees ($t$ has only one node), then $S(t) := 0$.

If $t$ has subtrees $t_1, \ldots, t_n$, then let $k := \max\{S(t_1), \ldots, S(t_n)\}$.

If exactly one subtree of $t$ has Strahler number $k$, then $S(t) := k$; otherwise, $S(t) := k + 1$. 

```
1
  0 1
  0 1
  0 0

2
  1 1
  0 0

3
  2
  1 1
  0 0

2
  1 1
  0 0

0 1 0 1 0 0 1 0
  0 0 0 0 0 1
  0 0
```
Understanding Strahler numbers

A tree has Strahler number \( k > 0 \) if it consists of a spine

- with subtrees of Strahler number at most \( k - 1 \)
- ending at a node with two subtrees of dimension exactly \( k - 1 \).
Understanding Strahler numbers

A binary tree $T$ has Strahler number $k > 0$ if it consists of a spine

- with subtrees of Strahler number at most $k - 1$
- ending at subtrees of dimension exactly $k - 1$. 

![Diagram](image-url)
Characterizations of the Strahler number

**Fact:** The Strahler number of a tree is the height of the largest minor that is a full binary tree.

**Fact:** The Strahler number of an arithmetic expression is the minimal number of registers needed to evaluate it.

```
R_1 ← y
R_2 ← z
R_2 ← R_1 × R_2
R_2 ← x
R_1 ← R_1 + R_2
R_2 ← w
R_1 ← R_1 × R_2
```

```
+   W
   / \
  x   ×
 /   /
y   z
```
Computing the $k$-th Newton approximant

\[ G : X \rightarrow aXX \mid bX \mid c \]

Define grammars $G_0, G_1, G_2 \ldots$ such that:

trees of $G_k \equiv$ trees of $G$ of Strahler number $\leq k$
Computing the $k$-th Newton approximant

$$G : X \rightarrow aXX | bX | c$$

Define grammars $G_0, G_1, G_2 \ldots$ such that:

trees of $G_k = \text{trees of } G \text{ of Strahler number } \leq k$

Idea: $X_{\langle k \rangle}$ generates the trees of Strahler number $= k$

$X_{[k]}$ generates the trees of Strahler number $\leq k$

$$G_k : X_{[k]} \rightarrow X_{\langle k \rangle} | X_{[k-1]}$$

$$X_{\langle k \rangle} \rightarrow aX_{\langle k-1 \rangle}X_{\langle k-1 \rangle} | aX_{[k-1]}X_{\langle k \rangle} | aX_{\langle k \rangle}X_{[k-1]} | bX_{\langle k \rangle}$$

$$\ldots$$

$$G_1 : X_{[1]} \rightarrow X_{\langle 1 \rangle} | X_{[0]}$$

$$X_{\langle 1 \rangle} \rightarrow aX_{[0]}X_{[0]} | aX_{[1]}X_{[0]} | aX_{[0]}X_{[1]} | bX_{\langle 1 \rangle}$$

$$G_0 : X_{[0]} \rightarrow c$$
Computing the $k$-th Newton approximant

\[ G : X \rightarrow aXX | bX | c \]

Define grammars $G_0, G_1, G_2 \ldots$ such that:

\text{trees of } G_k = \text{trees of } G \text{ of Strahler number } \leq k

Idea: $X_{\langle k \rangle}$ generates the trees of Strahler number $= k$

$X_{[k]}$ generates the trees of Strahler number $\leq k$

\[
X_{[k]} = X_{\langle k \rangle} + X_{[k-1]} \\
X_{\langle k \rangle} = aX_{\langle k-1 \rangle}X_{\langle k-1 \rangle} + aX_{[k-1]}X_{\langle k \rangle} + aX_{\langle k \rangle}X_{[k-1]} + bX_{\langle k \rangle} \\
\ldots \\
X_{[1]} = X_{\langle 1 \rangle} + X_{[0]} \\
X_{\langle 1 \rangle} = aX_{[0]}X_{[0]} + aX_{[1]}X_{[0]} + aX_{[0]}X_{[1]} + bX_{\langle 1 \rangle} \\
X_{[0]} = c\]
Computing the \( k \)-th Newton approximant

The equation

\[
X_{\langle k \rangle} = aX_{\langle k-1 \rangle}X_{\langle k-1 \rangle} + aX_{[k-1]}X_{\langle k \rangle} + aX_{\langle k \rangle}X_{[k-1]} + bX_{\langle k \rangle}
\]

is linear in \( X_{\langle k \rangle} \).
Computing the $k$-th Newton approximant

The equation

$$X_{(k)} = aX_{(k-1)}X_{(k-1)} + aX_{[k-1]}X_{(k)} + aX_{(k)}X_{[k-1]} + bX_{(k)}$$

is linear in $X_{(k)}$. 
Computing the $k$-th Newton approximant

The equation

$$X_{(k)} = aX_{(k-1)}X_{(k-1)} + aX_{[k-1]}X_{(k)} + aX_{(k)}X_{[k-1]} + bX_{(k)}$$

is linear in $X_{(k)}$.

Newton’s method approximates the solution of non-linear equations assuming one can solve (or approximate) linear equations.
Computing the $k$-th Newton approximant

The equation

$$X_{\langle k \rangle} = aX_{\langle k-1 \rangle} X_{\langle k-1 \rangle} + aX_{[k-1]} X_{\langle k \rangle} + aX_{\langle k \rangle} X_{[k-1]} + bX_{\langle k \rangle}$$

is linear in $X_{\langle k \rangle}$.

Newton’s method approximates the solution of non-linear equations assuming one can solve (or approximate) linear equations.

Over commutative semirings:

$$X_{\langle k \rangle} = (2aX_{[k-1]} + b)X_{\langle k \rangle} + aX_{\langle k-1 \rangle} X_{\langle k-1 \rangle}$$

$$X_{\langle k \rangle} = (2aX_{[k-1]} + b)^* a X_{\langle k-1 \rangle}^2$$
Some applications
Termination proofs

Theorem: Let $X = f(X)$ be a system of $n$ polynomial fixpoint equations over an idempotent and commutative semiring. Then Newton’s method terminates after at most $n + 1$ iterations.

Idempotent: number of copies of each tree is irrelevant

Commutative: trees with different order of leaves have the same yield.

Prove that for every derivation tree there is a derivation tree of Strahler number at most $n + 1$ with the same leaves, possibly in different order
Parikh’s theorem

Theorem [Parikh ’66]: For every context-free language there is a regular language with the same commutative image.

Problem: Given a CFG $G$, construct an automaton $A$ such that $L(G)$ and $L(A)$ have the same commutative image.

Solution: Use that $L(G)$ and $L(G^n)$ have the same commutative image.

Construct $A$ whose runs “simulate” the derivations of $G_n$. 
Parikh’s theorem

Example: \[ A_1 \rightarrow A_1 A_2 \mid a \quad A_2 \rightarrow bA_2 aA_2 \mid cA_1 \]

\[
\begin{align*}
A_1 & \quad \rightarrow \quad A_1 A_2 \\
\Rightarrow & \quad \rightarrow \quad A_1 bA_2 aA_2 \\
\Rightarrow & \quad \rightarrow \quad A_1 bcA_1 aA_2 \\
\Rightarrow & \quad \rightarrow \quad abcA_1 aA_2 \\
\Rightarrow & \quad \rightarrow \quad abcaaA_2 \\
\Rightarrow & \quad \rightarrow \quad abcaacA_1 \\
\Rightarrow & \quad \rightarrow \quad abcaaca
\end{align*}
\]

Graph representation:

- States: 0, 1, 2, 3
- Transitions:
  - \( A_1 \rightarrow \epsilon \rightarrow (0, 1) \)
  - \( A_1 \rightarrow ba \rightarrow (1, 1) \)
  - \( A_1 \rightarrow c \rightarrow (2, 1) \)
  - \( A_1 \rightarrow a \rightarrow (1, 1) \)
  - \( A_1 \rightarrow a \rightarrow (0, 1) \)
  - \( A_1 \rightarrow c \rightarrow (1, 0) \)
  - \( A_1 \rightarrow a \rightarrow (0, 0) \)
  - \( A_1 \rightarrow b \rightarrow (1, 0) \)
  - \( A_1 \rightarrow a \rightarrow (0, 0) \)

Arrows indicate transitions from one state to another.
A recommendation system

Idea: design a recommendation system in which:

- individuals can recommend other individuals or groups;
- membership in groups is defined in a fuzzy quantitative way;
- groups can be defined recursively (the friends of my friends are my friends)

Participants: researchers, universities, conferences, papers . . .

Relations: researcher-of, professor-at, student-of, author-of, . . .

- Notation: $p.r$
- Meaning: group of participants that are in relation $r$ with $p$. 
Rules

Information about group membership expressed through membership rules

Aachen.professor ← Grädel
Aachen.researcher ← Aachen.professor
Aachen.researcher ← Aachen.researcher.phd-student
Grädel.phd-student ← Naaf
To establish that Naaf is a researcher at Aachen:

\[
\text{Aachen.researcher} \leftarrow \text{Aachen.researcher.phd-student}
\]
To establish that Naaf is a researcher at Aachen:

\[
\text{Aachen.researcher} \leftarrow \text{Aachen.researcher.phd-student} \\
\leftarrow \text{Aachen.professor.phd-student}
\]
Inference

To establish that Naaf is a researcher at Aachen:

Aachen.researcher ← Aachen.researcher.phd-student
← Aachen.professor.phd-student
← Grädel.phd-student
To establish that Naaf is a researcher at Aachen:

\[
\text{Aachen.researcher} \leftarrow \text{Aachen.researcher.phd-student}
\]
\[
\leftarrow \text{Aachen.professor.phd-student}
\]
\[
\leftarrow \text{Grädel.phd-student}
\]
\[
\leftarrow \text{Naaf}
\]
Weights

Membership rules carry weights (degree of membership)

\[
\begin{align*}
\text{LICS.author} & \overset{14/1000}{\leftarrow} \text{Grädel} \\
\text{LICS.author} & \overset{1/1000}{\leftarrow} \text{Naaf}
\end{align*}
\]
Weights

Membership rules carry weights (degree of membership)

\[
\begin{align*}
\text{LICS.author} & \leftarrow \frac{14}{1000} \quad \text{Grädel} \\
\text{LICS.author} & \leftarrow \frac{1}{1000} \quad \text{Naaf} \\
\text{Grädel.co-author} & \leftarrow \frac{1}{30} \quad \text{Naaf} \\
\text{Naaf.co-author} & \leftarrow \frac{1}{2} \quad \text{Grädel}
\end{align*}
\]
Weights

Membership rules carry weights (degree of membership)

\[
\begin{align*}
\text{LICS.author} & \leftarrow \frac{14}{1000} \quad \text{Grädel} \\
\text{LICS.author} & \leftarrow \frac{1}{1000} \quad \text{Naaf} \\
\text{Grädel.co-author} & \leftarrow \frac{1}{30} \quad \text{Naaf} \\
\text{Naaf.co-author} & \leftarrow \frac{1}{2} \quad \text{Grädel}
\end{align*}
\]

Recursive group definitions with damping factors.

\[
\begin{align*}
\text{Grädel.community} & \leftarrow \frac{1}{2} \quad \text{Grädel.co-author} \\
\text{Grädel.community} & \leftarrow 0.5 \quad \text{Grädel.community.co-author}
\end{align*}
\]
Recommending individuals and groups

Recommendations

Grädel $\leftarrow 10$ Naaf
Grädel $\leftarrow 8$ LICS.author

Grädel recommends Naaf through two paths

Grädel $\leftarrow 10$ Naaf
Grädel $\leftarrow 8$ LICS.author $\leftarrow 1/1000$ Naaf

Semiring operations $\oplus$ and $\odot$ to aggregate values:

Grädel $\leftarrow 10 \oplus (8 \odot 1/30)$ Naaf
Computing reputation

Assume a given set of weighted rules and recommendations

**Reputation of an individual**: total weight with which participants recommend the individual (through all possible paths)

**Theorem**: The reputation of the individuals is the least fixpoint of a system of non-linear equations
Thank you for your attention