Newton's method extended to ω-continuous semi-rings

Prathmesh Prabhu

Taedong Kim

Srikrishna Sridhar

Project Goal

- Try out Newton's method extended to semi-rings to solve program analysis problems[1].

[1] Esparza et. al "Newton's method on ω continuous semi-rings"

Outline

- Background & Motivation
- Approach
  - Past Work
  - An alternate approach
- Results
  - Discussion
  - Future directions

BACKGROUND & MOTIVATION

Problem Statement

Compute the least solution of a system of equations defined on an ω−continuous semi-ring.

$$\vec{x} = \vec{F}(\vec{x})$$

The oh-my-god slide.

A semi-ring is an algebraic structure: (S,$\oplus$,$\otimes$)

$\oplus$ $\otimes$ are internal compositions on S with the following properties -

$(a \oplus b) \oplus c = a \oplus (b \oplus c)$ ($\oplus$ assoc.)

$a \oplus b = b \oplus a$ ($\oplus$ commutative)

$0 \oplus a = a = a \oplus 0$ (additive identity)

$(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$

$0 \otimes a = 0 = a \otimes 0$ (0 annihilates $\otimes$)

$a \oplus a = a$ (optional idempotency)

$(a \otimes b) \otimes c = a \otimes (b \otimes c)$ ($\otimes$ assoc.)

$a \otimes b = b \otimes a$ (optional $\otimes$ commutative)

$1 \otimes a = a = a \otimes 1$ (extend identity)

$c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$

Semi-rings – Intuition

$$(S, \otimes, \oplus)$$

Semi-rings – Intuition

$$(S, \otimes, \oplus)$$

- Let's take $\mathbb{R}$

$\oplus$: +

$\otimes$: x

Semi-rings – Intuition

$$(S, \otimes, \oplus)$$

- Let's take $\mathbb{R}$

$\oplus$: +

$\otimes$: x

- $\otimes$ not commutative

Semi-rings – Intuition

$$(S, \otimes, \oplus)$$

- Let's take $\mathbb{R}$

$\oplus$: + $\otimes$: x

- $\otimes$ not commutative
- No '-' operation

Semi-rings – Intuition

$$(S, \otimes, \oplus)$$

- Let's take $\mathbb{R}$

$\oplus$: + $\otimes$: x

- $\otimes$ not commutative
- No '-' operation
- No '/' operation

Semi-rings vis-à-vis Lattices

1. Join semilattices are the same as semi-rings -

$$\vee : \oplus$$

Semi-rings vis-à-vis Lattices

1. Join semilattices are the same as semi-rings -

$$\vee : \oplus$$

    1. Program semantics

Semi-rings vis-à-vis Lattices

1. Join semilattices are the same as semi-rings -

$$\vee : \oplus$$

1. Program semantics
2. Abstract interpretation

Semi-rings vis-à-vis Lattices

1. Join semilattices are the same as semi-rings -

$$\vee : \oplus$$

   1. Program semantics
   2. Abstract interpretation

2. Many useful monotonic functions that arise in program analysis can be written using $\otimes$

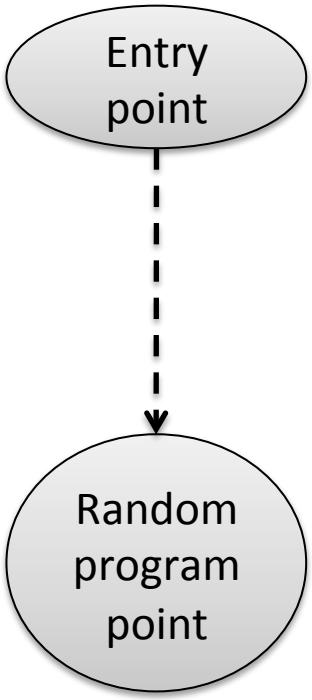
Semi-rings are useful !

Some well studied problems:

Semi-rings are useful !

Some well studied problems:

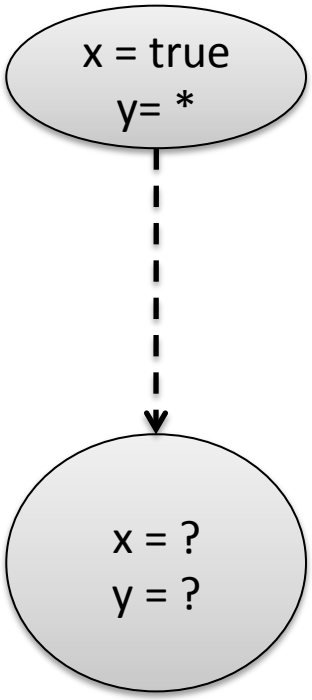
- Reachability Analysis



Semi-rings are useful !

Some well studied problems:

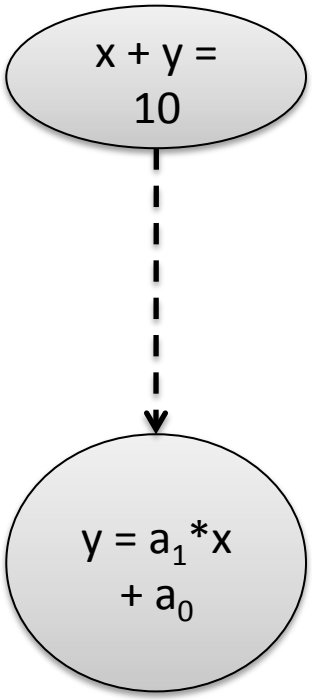
- Reachability Analysis
- Boolean Relational Analysis



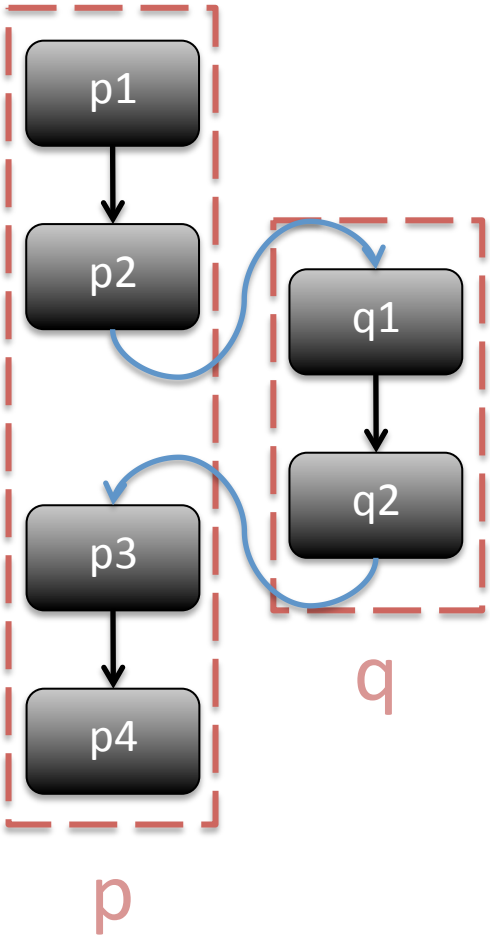
Semi-rings are useful !

Some well studied problems:

- Reachability Analysis
- Boolean Relational Analysis
- Affine Relations Analysis (ARA)

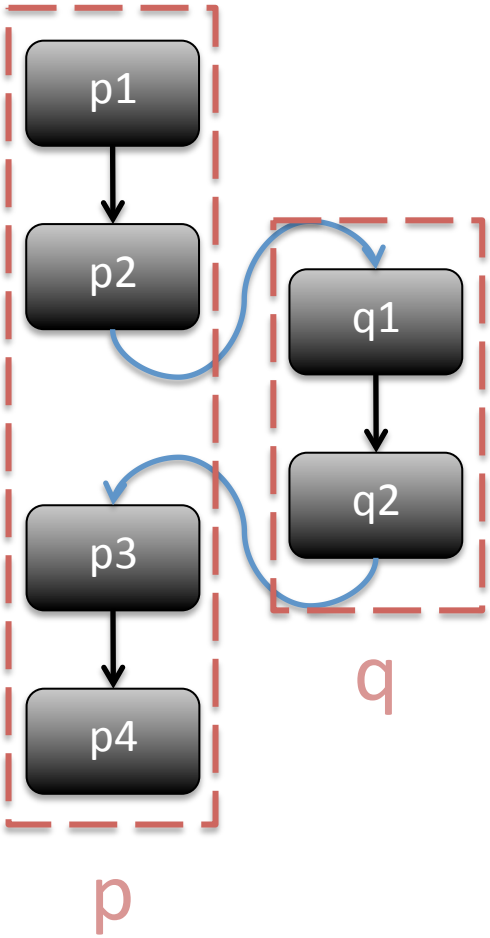


The reachability toy example



Write a system of equations, whose fixed point solution is exactly the set of reachable nodes in this CFG:

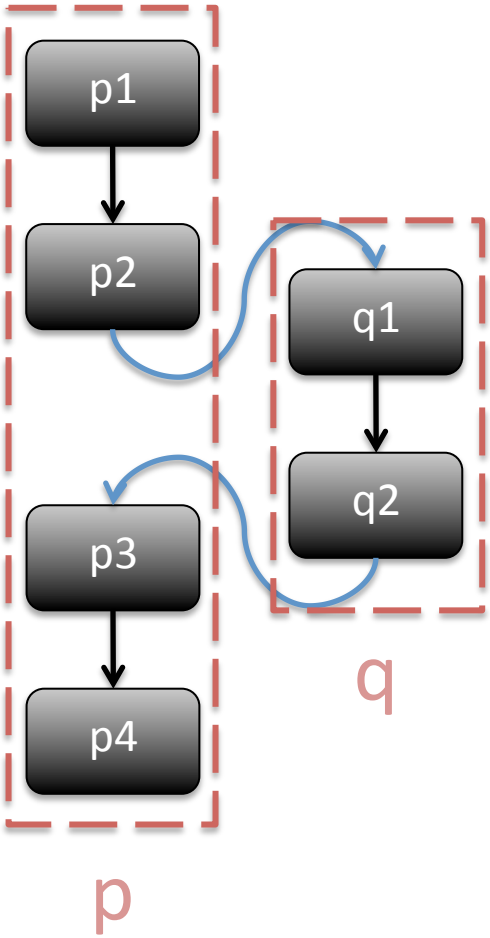
The reachability toy example



Write a system of equations, whose fixed point solution is exactly the set of reachable nodes in this CFG:

S = { r, n }

The reachability toy example

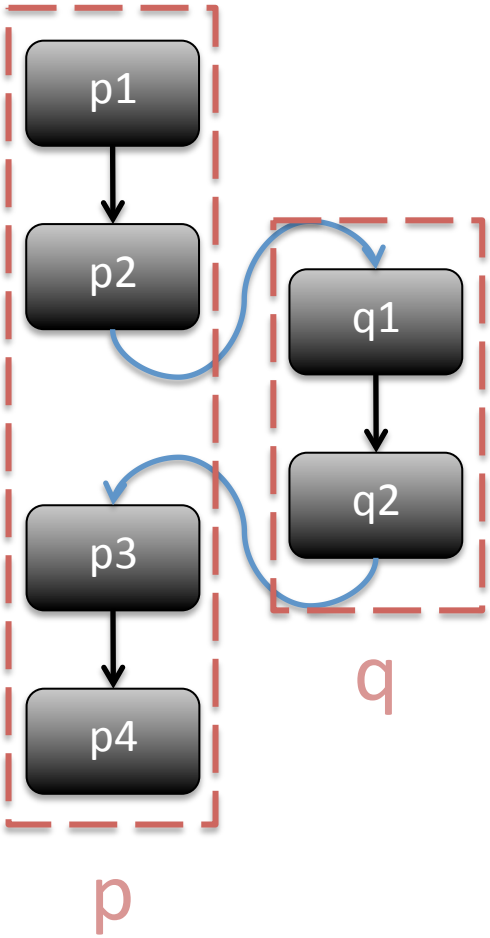


Write a system of equations, whose fixed point solution is exactly the set of reachable nodes in this CFG:

$$S = \{ r, n \}$$

Variables: p1, p2, p3, p4, q1, q2

The reachability toy example



Write a system of equations, whose fixed point solution is exactly the set of reachable nodes in this CFG:

S = { r, n }

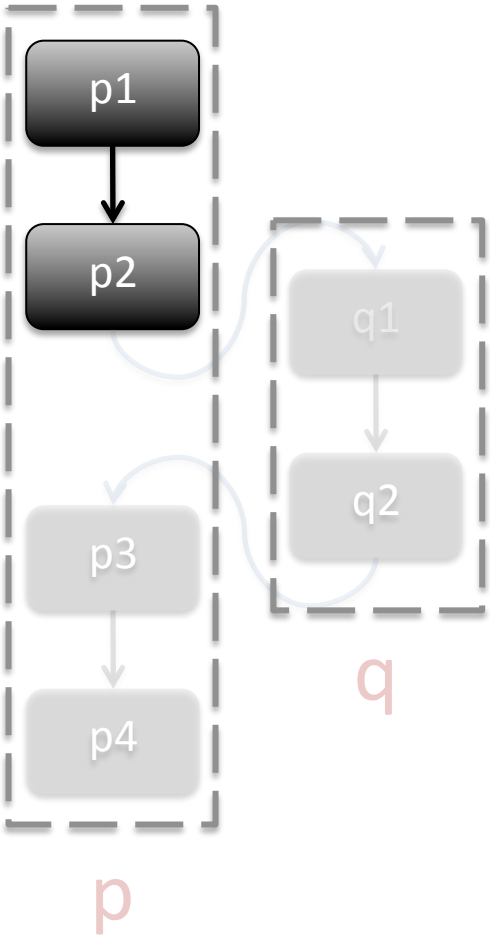
Variables: p1, p2, p3, p4, q1, q2

Initial Values:

p1 = r

p2,p3,p4,q1,q2 = n

The reachability toy example



Write a system of equations, whose fixed point solution is exactly the set of reachable nodes in this CFG:

$$S = \{ r, n \}$$

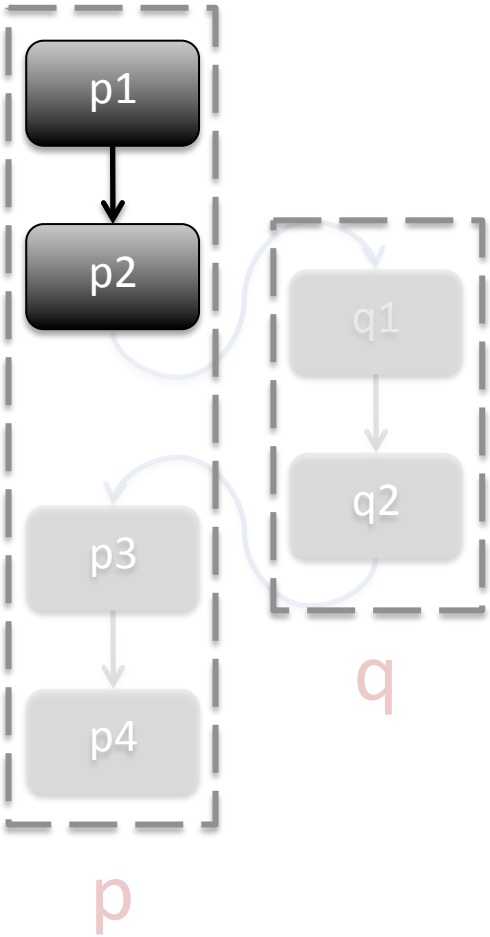
Variables: p1, p2, p3, p4, q1, q2

Initial Values:

p1 = r

p2,p3,p4,q1,q2 = n

The reachability toy example



Write a system of equations, whose fixed point solution is exactly the set of reachable nodes in this CFG:

S = { r, n }

Variables: p1, p2, p3, p4, q1, q2

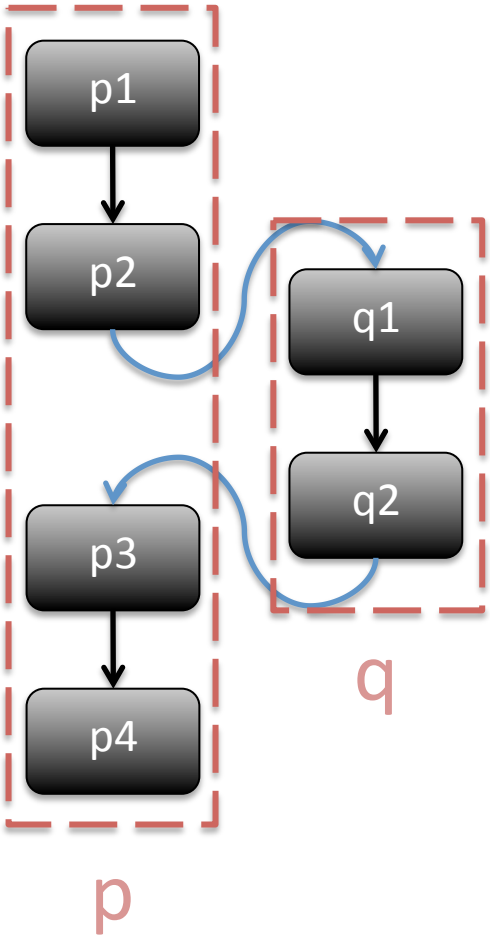
Initial Values:

p1 = r

p2,p3,p4,q1,q2 = n

$$p2 = p1 \otimes r$$

The reachability toy example



Write a system of equations, whose fixed point solution is exactly the set of reachable nodes in this CFG:

S = { r, n }

Variables: p1, p2, p3, p4, q1, q2

Initial Values:

p1 = r

p2,p3,p4,q1,q2 = n

$$
\begin{aligned}
p2 &= p1 \otimes r \\
q1 &= p2 \otimes r \\
q2 &= q1 \otimes r \\
p3 &= p2 \otimes q2 \\
p4 &= p3 \otimes r
\end{aligned}
$$

Past and Present

APPROACH

How do we solve such systems?

$$p2 = p1 \otimes r$$
$$q1 = p2 \otimes r$$
$$q2 = q1 \otimes r$$
$$p3 = p2 \otimes q2$$
$$p4 = p3 \otimes r$$

How do we solve such systems?

$$
\begin{aligned}
p2 &= p1 \otimes r \\
q1 &= p2 \otimes r \\
q2 &= q1 \otimes r \\
p3 &= p2 \otimes q2 \\
p4 &= p3 \otimes r
\end{aligned}
\quad \longrightarrow \quad \vec{x} = \mathbb{F}(\vec{x})
$$

Solving $\vec{x} = \mathbb{F}(\vec{x})$

Some useful properties

Solving $\vec{x} = \mathbb{F}(\vec{x})$

Some useful properties

- Domain of X (semi-ring) is a partial order.

Solving $\vec{x} = \mathbb{F}(\vec{x})$

Some useful properties

- Domain of X (semi-ring) is a partial order.
- F is continuous and monotonic.

Solving $\vec{x} = \mathbb{F}(\vec{x})$

Some useful properties

- Domain of X (semi-ring) is a partial order.
- F is continuous and monotonic.

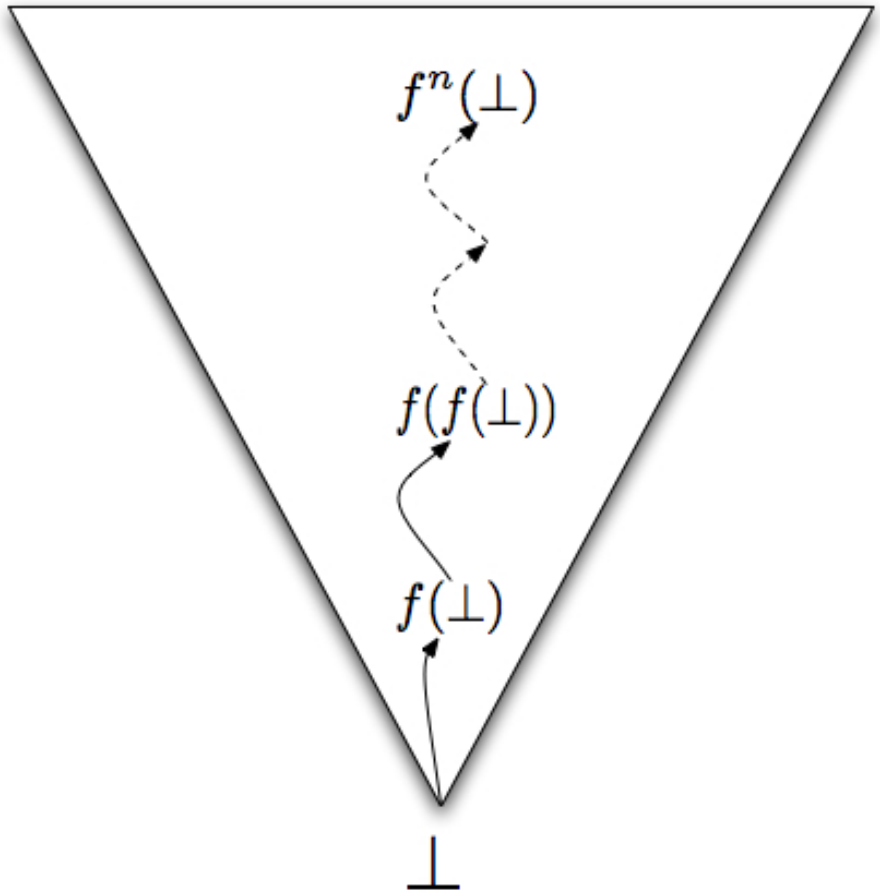
Hence, we can apply a technique called Kleene's Method.

Kleene's Iteration Scheme

- Start from bottom and recursively apply F

$$\bot \leq f(\bot) \leq f(f(\bot)) \leq \ldots \leq f^n(\bot) \leq \ldots$$

Kleene's Iteration Scheme



Kleene's Iteration Scheme

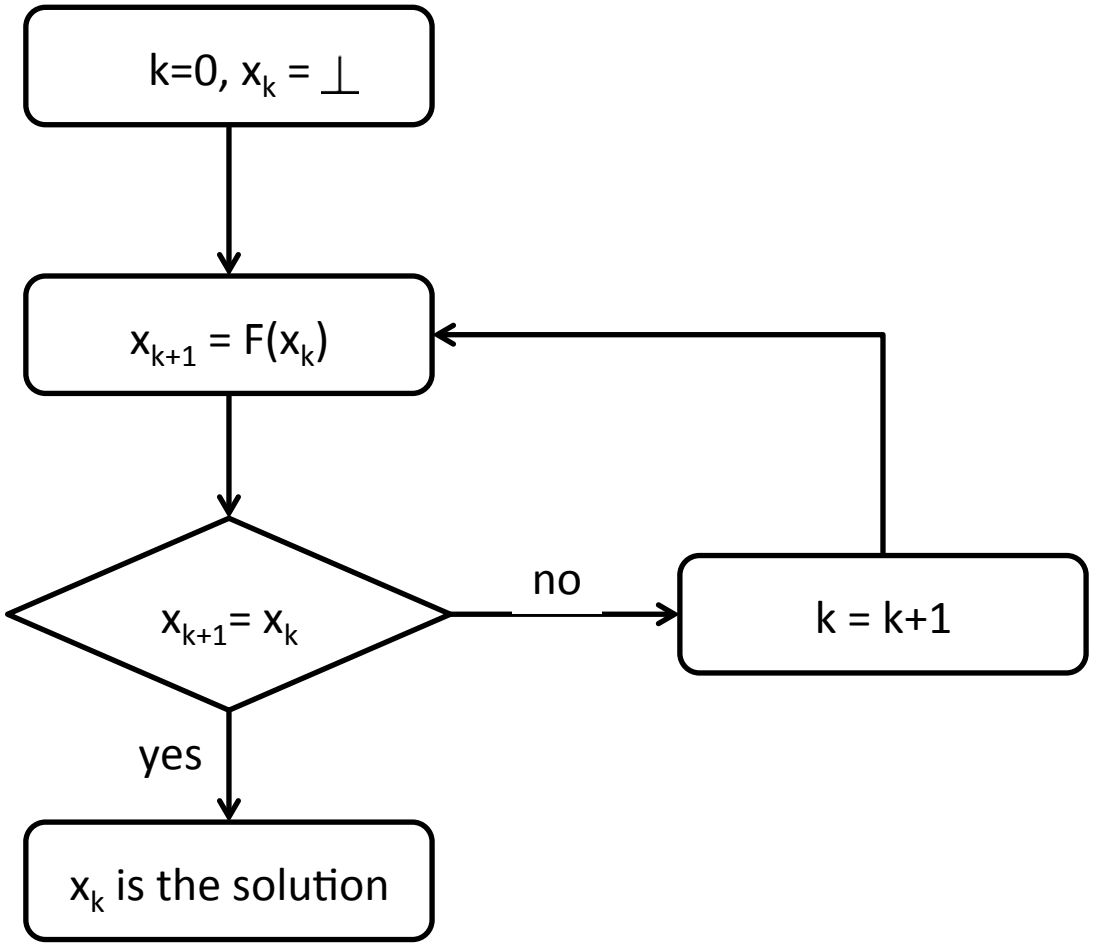
- Start from bottom and recursively apply F

$$\perp \leq f(\perp) \leq f(f(\perp)) \leq \ldots \leq f^n(\perp) \leq \ldots$$

- Series approaches least fixed point(lfp).

$$\mathrm{lfp}(f) = \sup(f^n(\perp) \mid n \in \mathcal{N})$$

Kleene's Iteration Scheme



Drawbacks

- Kleene's iteration scheme converges very slowly.

Another approach..

- Recently, Newton's method has been extended to functions on semi-ring domains.[1]

[1] Esparza et. al "Newton's method on ω continuous semi-rings"

Another approach..

- Recently, Newton's method has been extended to functions on semi-ring domains.

Intuition

- In real domain

$$x = F(x) \;\Leftrightarrow\; x - F(x) = 0$$

If it’s fast, it’s Newton

- The Newton’s method is the most common technique to solve a root-finding problem.

If it's fast, it's Newton

- The Newton's method is the most common technique to solve a root-finding problem.
- Quadratic convergence
  - Error term reduces quadratically.

$$\frac{||x_{k+1} - x^*||}{||x_k - x^*||^2} \leq r$$

If it's fast, it's Newton

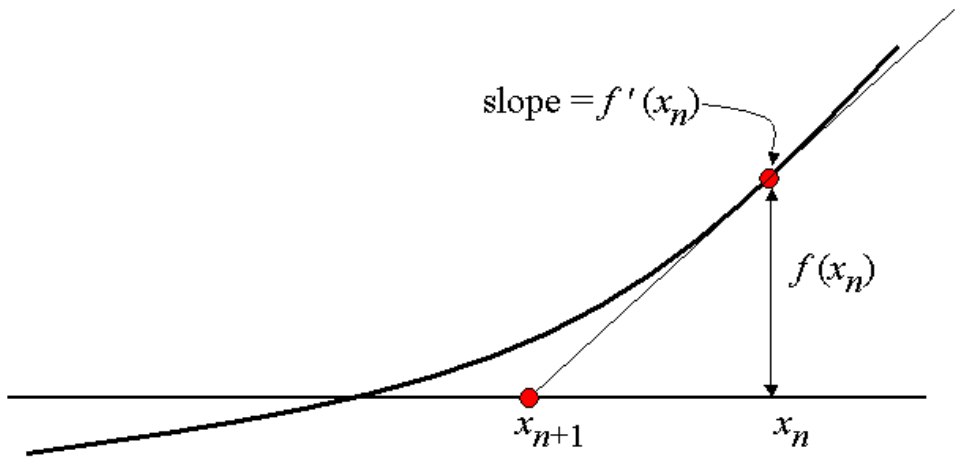
- The Newton's method is the most common technique to solve a root-finding problem.
- Quadratic convergence
  - Error term reduces quadratically.

$$\frac{||x_{k+1} - x^*||}{||x_k - x^*||^2} \leq r$$

- Newton's methods is robust on semi-rings.

Newton's method

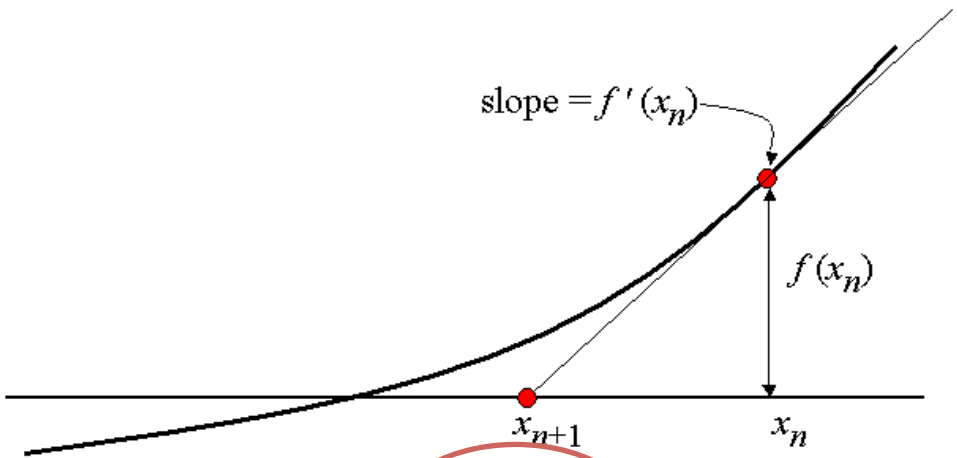
- Start with an initial guess of root of the function.
- Compute a linear approximation at the point.
- The x-intercept of the new point is a better approximation to the root.



$$x_{n+1} = x_n - \frac{f'(x_n)}{f(x_n)}$$

Newton's method

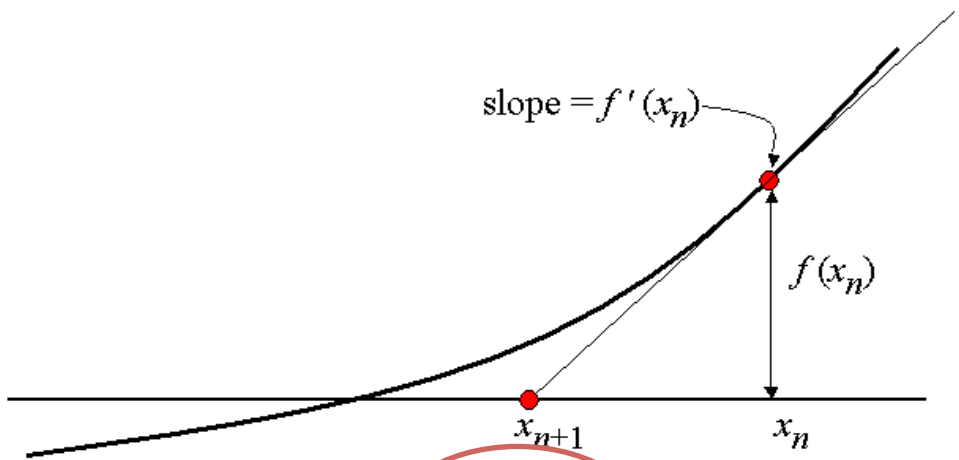
- Start with an initial guess of root of the function.
- Compute a linear approximation at the point.
- The x-intercept of the new point is a better approximation to the root.



$$x_{n+1} = x_n - \frac{f'(x_n)}{f(x_n)}$$

Newton's method

- As we saw, there is no subtraction(-) and division(/) on the semi-rings.
- Esparza et al. redefines linearization on semi-rings.



$$x_{n+1} = x_n - \frac{f'(x_n)}{f(x_n)}$$

Linearization

Linearization of $f(X) = X$ at $X = X_0$

is defined as $Df_{X_0}(X) + f(X_0) = X.$

Linearization

Linearization of $f(X) = X$ at $X = X_0$

is defined as $Df_{X_0}(X) + f(X_0) = X.$

Linearizing $p_2 \otimes q_2 = p_3$ at $(p_2, q_2) = (n, n)$?

Linearization

Linearization of $f(X) = X$ at $X = X_0$

is defined as $Df_{X_0}(X) + f(X_0) = X.$

Linearizing $p_2 \otimes q_2 = p_3$ at $(p_2, q_2) = (n, n)$?

$$(n \otimes q2) \oplus (p_2 \otimes n) \oplus (n \otimes n) = p_3$$

Linearization

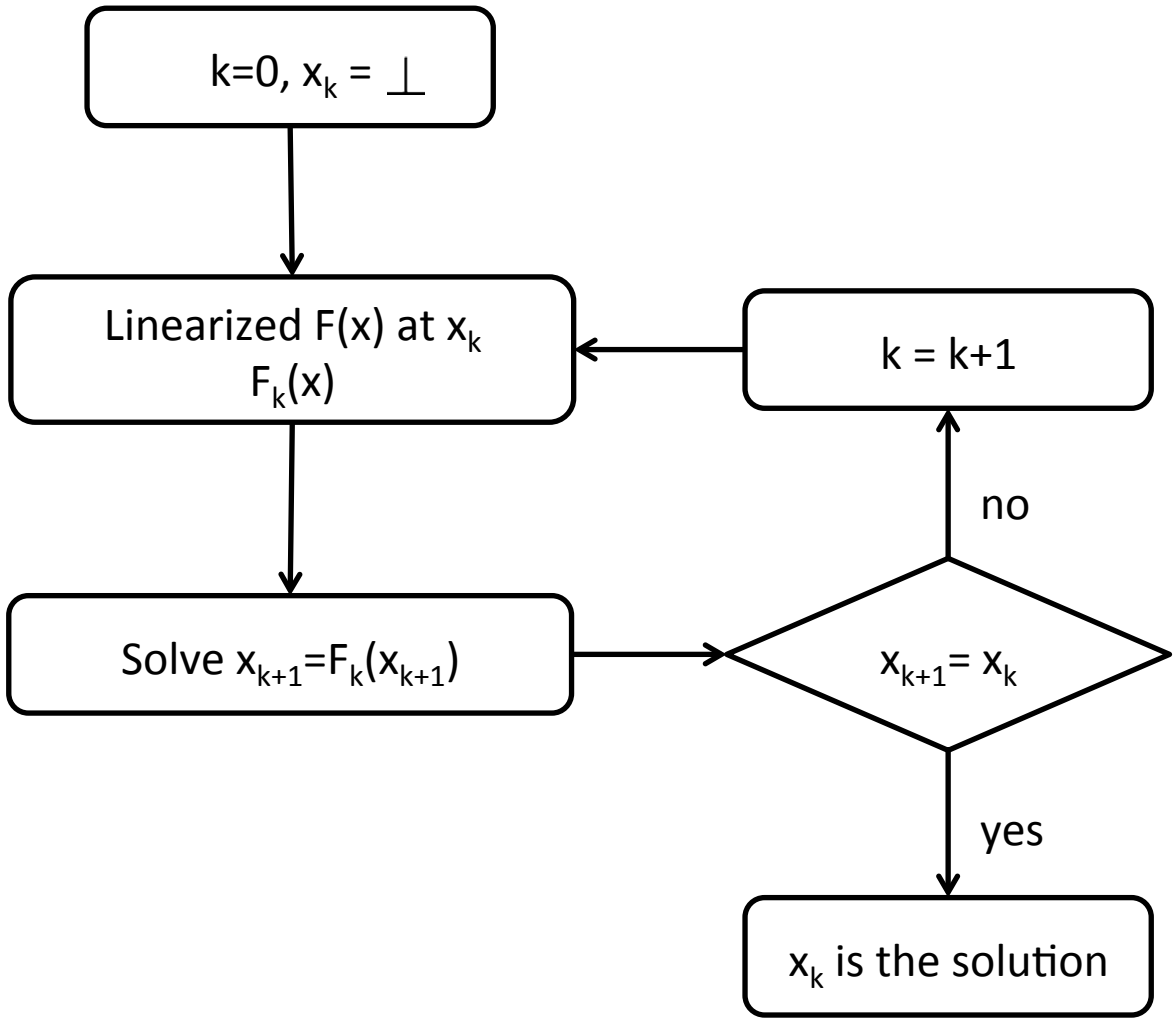
Linearization of $f(X) = X$ at $X = X_0$

is defined as $Df_{X_0}(X) + f(X_0) = X.$

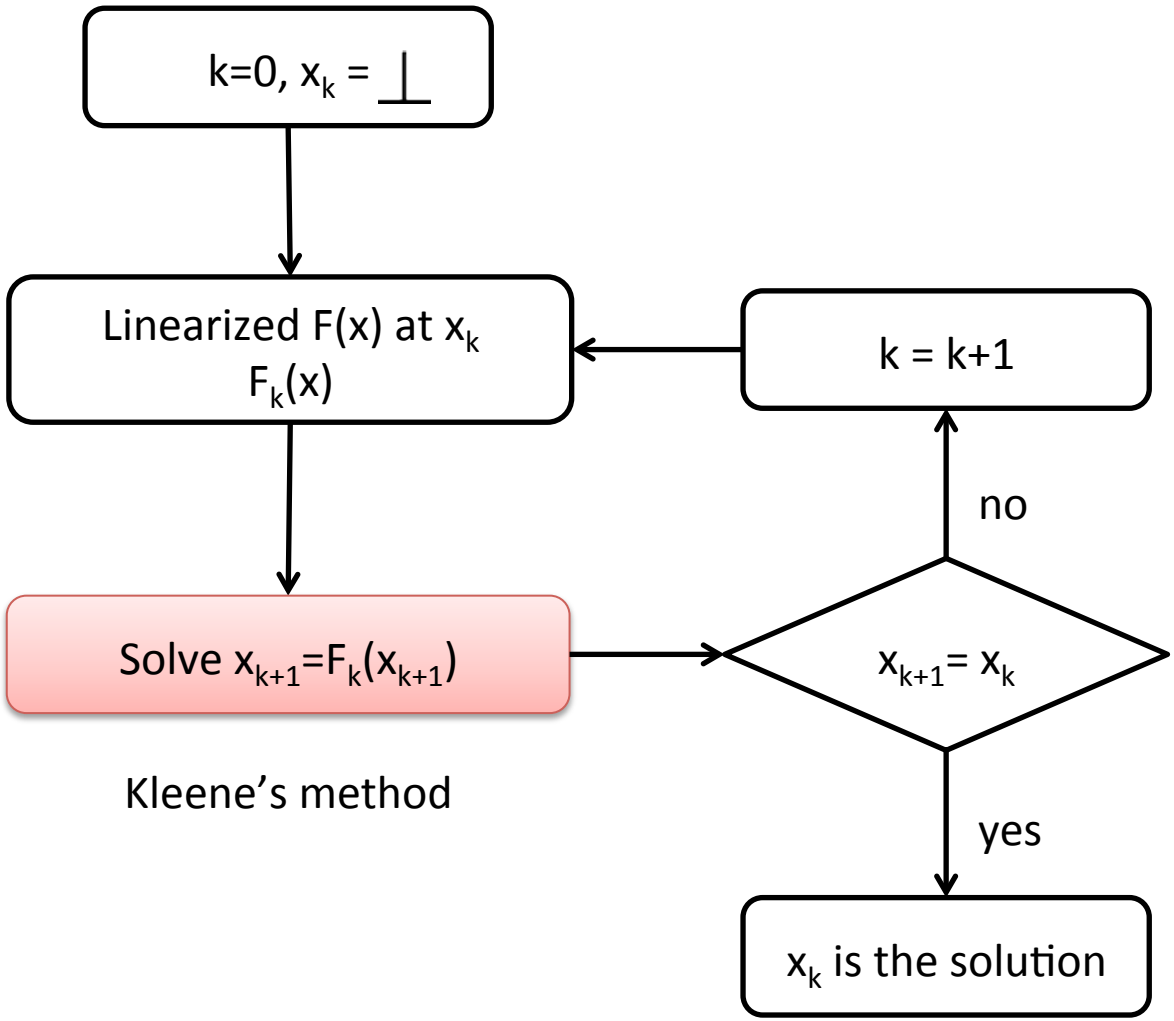
Linearizing $p_2 \otimes q_2 = p_3$ at $(p_2, q_2) = (n, n)$?

$$\underbrace{(n \otimes q2) \oplus (p_2 \otimes n)}_{Df_{X_0}(X)} \oplus \underbrace{(n \otimes n)}_{f(X_0)} = p_3$$

Newton's Iteration Scheme



Newton's Iteration Scheme



RESULTS & FUTURE DIRECTIONS

Experimental Results

- Verify correctness of the solvers for
  - Reachability Analysis
  - Boolean relational Analysis

Experimental Results

- Verify correctness of the solvers for
  - Reachability Analysis
  - Boolean relational Analysis

- Performance analysis for
  - Affine Relational Analysis

ARA Semi-Rings

- Test on Windows binaries of varying size.

ARA Semi-Rings

- Test on Windows binaries of varying size.

- Solve system of equations on Control flow graphs with transitions labeled with ARA weights.

ARA Semi-Rings

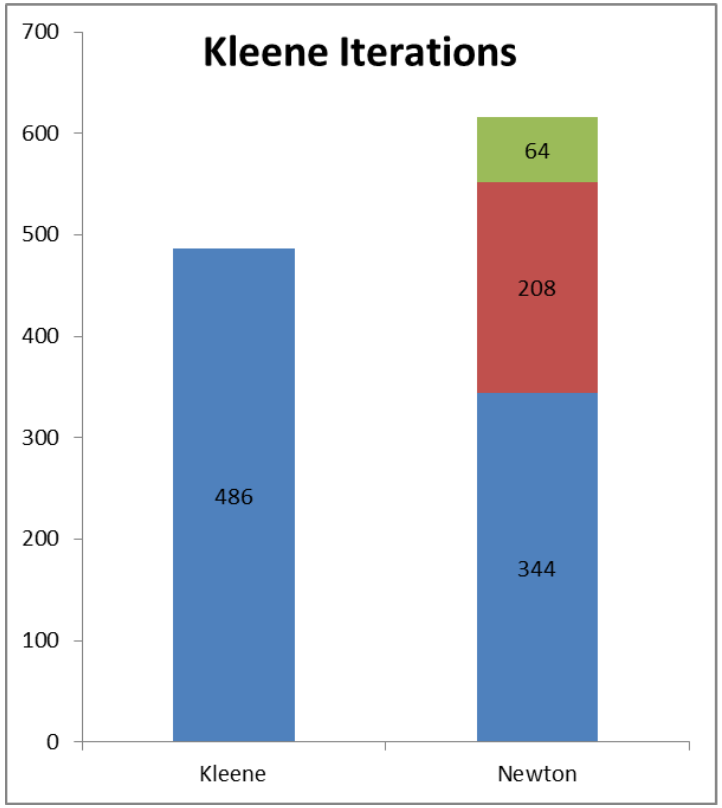
- Test on Windows binaries of varying size.
- Solve system of equations on Control flow graphs with transitions labeled with ARA weights.
- Solve this system using ***Kleene*** and ***Newton*** methods.

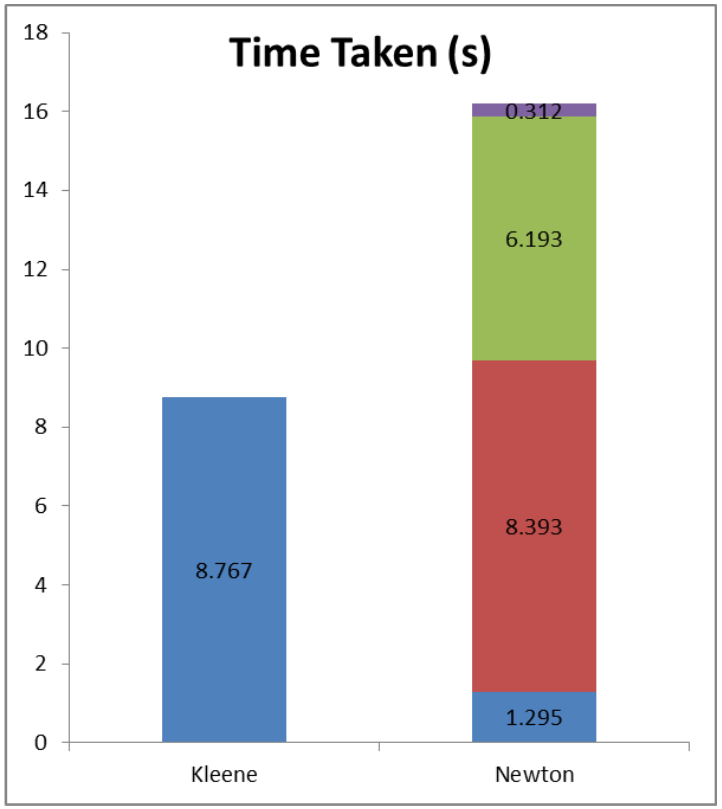
Benchmarks

- Number of Kleene iterations
  - *Note : For newton method, we add up Kleene iterations from all the Newton steps.*

- Time taken

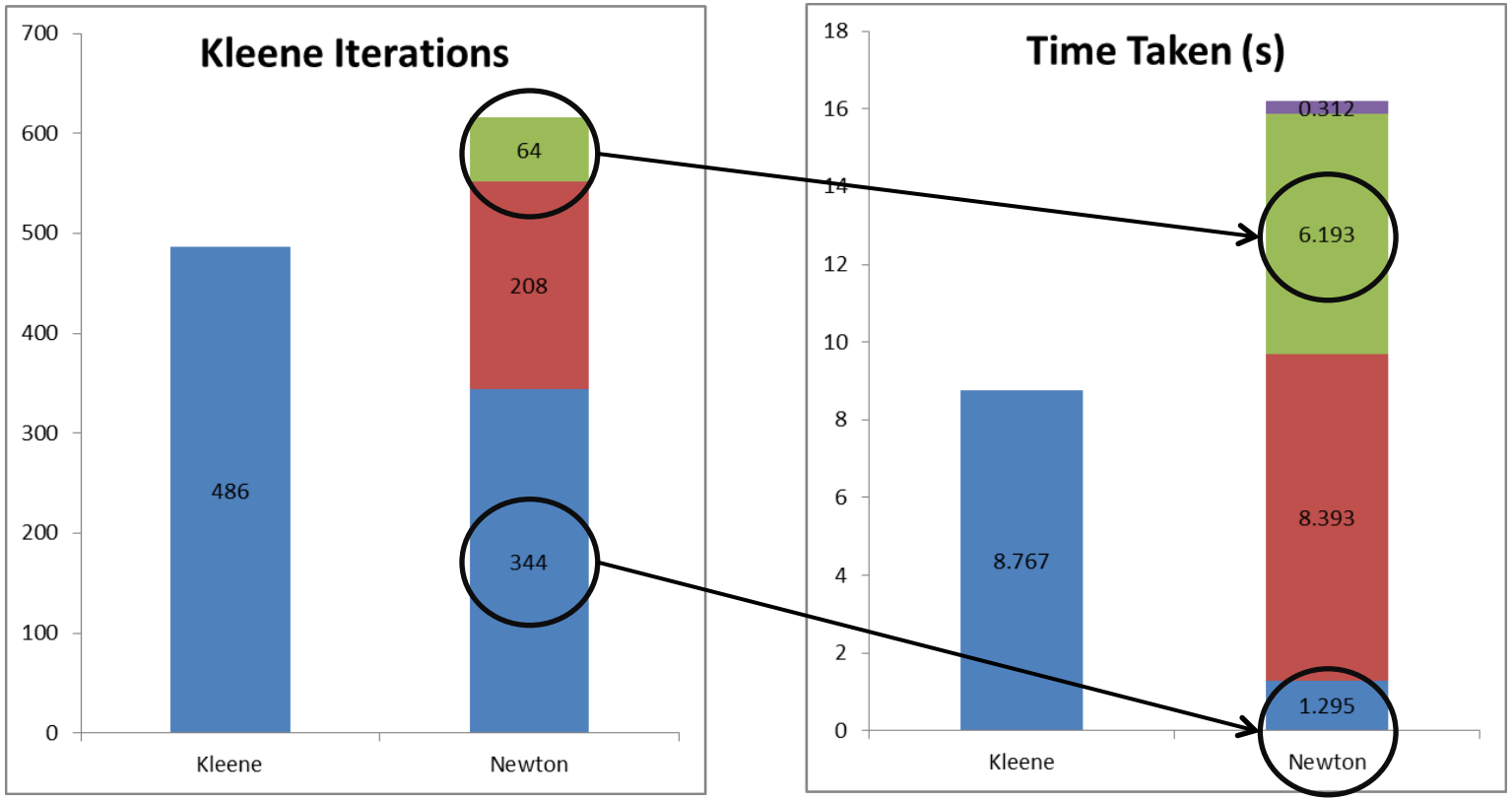
Time analysis





write (250)

Time analysis



write (250)

Keep it Simple Stupid!

$$p_3 = (n \otimes q2) \oplus (p_2 \otimes n) \oplus (n \otimes n)$$

Keep it Simple Stupid!

$$p_3 = (n \otimes q2) \oplus (p_2 \otimes n) \oplus (n \otimes n)$$

$$\Downarrow$$

$$p_3 = (n \otimes q2) \oplus p_3$$

$$p_3 = (p_2 \otimes n) \oplus p_3$$

Starting from $p_3 = n \otimes n$

Keep it Simple Stupid!

$$P_0 = f_{P_1}(P_1) + f_{P_2}(P_2) + f_{P_3}(P_3)$$

Keep it Simple Stupid!

$$P_0 = f_{P_1}(P_1) + f_{P_2}(P_2) + f_{P_3}(P_3)$$

$$\Downarrow$$

$$P_0 = f_{P_1}(P_1) + P_0$$

$$P_0 = f_{P_2}(P_2) + P_0$$

$$P_0 = f_{P_3}(P_3) + P_0$$

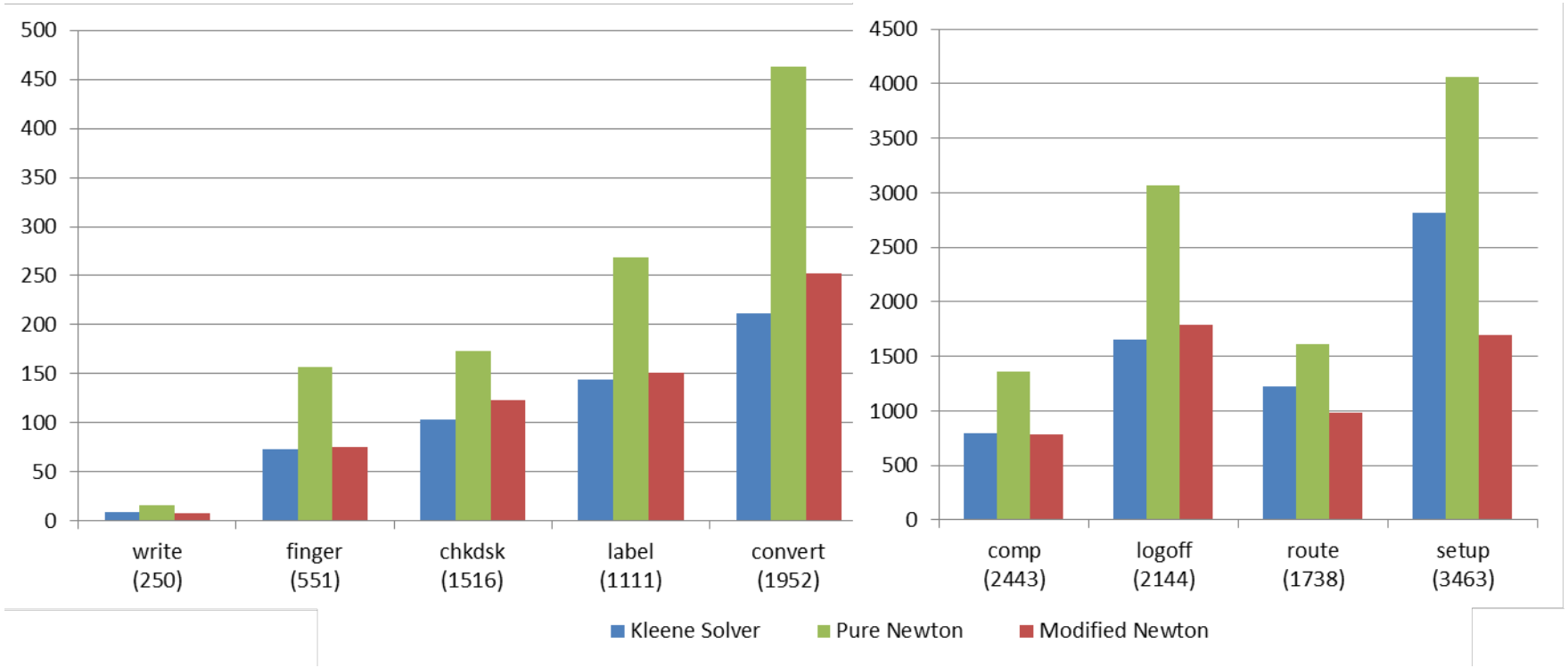
Results Summary

	Problem Size	Kleene Solver		Newton			Modified Newton		
		# of iter.	Time (s)	Newton Iter.	Kleene iter.	Time (s)	Newton Iter.	Kleene iter.	Time (s)
write	250	486	8.767	4	573	8.097	4	617	16.193
finger	551	1579	73.274	5	1623	75.831	5	1765	156.609
chkdsk	1516	4322	103.319	5	4733	123.428	5	5126	172.927
label	1111	3989	144.488	4	3996	150.774	5	4461	268.851
convert	1952	4937	211.318	5	5395	252.424	5	5921	463.352
comp	2443	9889	790.578	5	9653	786.772	5	10844	1360.4
logoff	2144	7769	1656.71	5	9514	1784.97	5	9488	3066.97
route	1738	6640	1228.81	5	6998	979.292	5	7193	1608.71
setup	3463	9684	2817.05	6	12711	1693.38	6	13235	4066.3

Results Summary

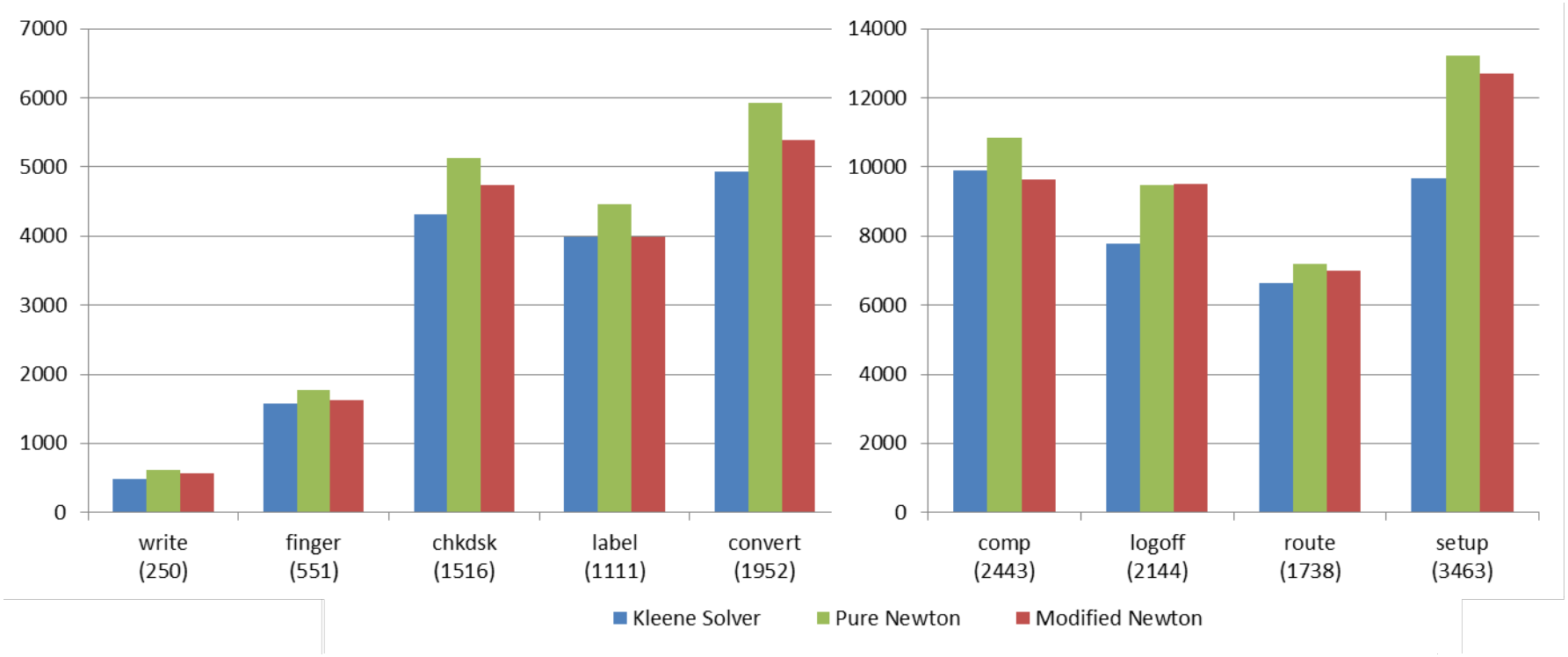
	Problem Size	Kleene Solver		Newton			Modified Newton		
		# of iter.	Time (s)	Newton Iter.	Kleene iter.	Time (s)	Newton Iter.	Kleene iter.	Time (s)
write	250	486	8.767	4	573	8.097	4	617	16.193
finger	551	1579	73.274	5	1623	75.831	5	1765	156.609
chkdsk	1516	4322	103.319	5	4733	123.428	5	5126	172.927
label	1111	3989	144.488	4	3996	150.774	5	4461	268.851
convert	1952	4937	211.318	5	5395	252.424	5	5921	463.352
comp	2443	9889	790.578	5	9653	786.772	5	10844	1360.4
logoff	2144	7769	1656.71	5	9514	1784.97	5	9488	3066.97
route	1738	6640	1228.81	5	6998	979.292	5	7193	1608.71
setup	3463	9684	2817.05	6	12711	1693.38	6	13235	4066.3

Run Newton run…



Time taken (sec) to converged

Basic steps ...



Number of Kleene iterations

Observations

- Number of newton steps is small.
- Low overhead between successive newton steps.
  - 98% time spent on solving linear systems

Future Directions

- Smarter Linear system solvers?
  - Incremental solvers.
  - Solvers that use the fact that systems are linear.

Thank you

Questions?

- Known unknowns
- Unknown unknowns