Polynomial Loops: Beyond Termination

23rd Conference on Logic for Programming, Artificial Intelligence and Reasoning

Marcel Hark  Florian Frohn  Jürgen Giesl
Motivation

Setting the stage

• Termination on a given input and Runtime Complexity are among the most important program properties.
  → Witnesses for non-termination indicate implementation bugs.
  → Runtime bounds indicate efficiency of a program.

• Drawback: Termination on a given input of general programs (Halting Problem) is undecidable [Turing 1937].
  → No technique which can (dis-)prove Halting Problem for all programs.

• Hope: Find sub-classes of programs where Halting Problem is decidable and runtime bounds can be computed.
  → For Halting Problem: Linear loops [Li '17, Kincaid et al. '19]

```while (ϕ)
    ⃗x ← A ⃗x
```

A has entries in \( \mathbb{R} \), \( ϕ \) a linear formula.

• What about non-linear behavior in condition/update?
• What about computability of runtime bounds?
Motivation

Setting the stage

- Termination on a given input and Runtime Complexity are among the most important program properties.
  - Witnesses for non-termination indicate implementation bugs.
  - Runtime bounds indicate efficiency of a program.
- Drawback: Termination on a given input of general programs (Halting Problem) is undecidable [Turing 1937].
  - No technique which can (dis-)prove Halting Problem for all programs.
- Hope: Find sub-classes of programs where Halting Problem is decidable and runtime bounds can be computed.
  - For Halting Problem: Linear loops [Li '17, Kincaid et al. '19]
  • \( \text{while } (\phi) \{ \vec{x} \leftarrow A \cdot \vec{x} \} \)
    - \( A \) has entries in \( \mathbb{R} \), \( \phi \) a linear formula.
- What about non-linear behavior in condition/update?
- What about computability of runtime bounds?
Motivation

Setting the stage

• Termination on a given input and Runtime Complexity are among the most important program properties.
  → Witnesses for non-termination indicate implementation bugs.

• Drawback: Termination on a given input of general programs (Halting Problem) is undecidable [Turing 1937].
  → No technique which can (dis-)prove Halting Problem for all programs.

• Hope: Find sub-classes of programs where Halting Problem is decidable and runtime bounds can be computed.
  → For Halting Problem: Linear loops [Li '17, Kincaid et al. '19]

  while (ϕ)
  {⃗x ← A·⃗x
  }

  A has entries in \( \mathbb{R} \)

• What about non-linear behavior in condition/update?
• What about computability of runtime bounds?
Motivation

Setting the stage

- Termination on a given input and Runtime Complexity are among the most important program properties.
  - Witnesses for non-termination indicate implementation bugs.
  - Runtime bounds indicate efficiency of a program.

- Drawback: Termination on a given input of general programs (Halting Problem) is undecidable [Turing 1937].
  - No technique which can (dis-)prove Halting Problem for all programs.

- Hope: Find sub-classes of programs where Halting Problem is decidable and runtime bounds can be computed.
  - For Halting Problem: Linear loops [Li '17, Kincaid et al. '19]

\[
\text{while } (\phi) \\
\vec{x} \leftarrow A \cdot \vec{x}
\]

- \(A\) has entries in \(R\), \(\phi\) a linear formula.

- What about non-linear behavior in condition/update?
- What about computability of runtime bounds?
**Motivation**

**Setting the stage**

- Termination on a given input and Runtime Complexity are among the most important program properties.
  - Witnesses for non-termination indicate implementation bugs.
  - Runtime bounds indicate efficiency of a program.

- **Drawback:**
  
- What about non-linear behavior in condition/update?
  
- What about computability of runtime bounds?
Motivation

Setting the stage

• Termination on a given input and Runtime Complexity are among the most important program properties.
  → Witnesses for non-termination indicate implementation bugs.
  → Runtime bounds indicate efficiency of a program.

• Drawback: Termination on a given input of general programs (Halting Problem) is undecidable [Turing 1937].
Motivation

Setting the stage

• Termination on a given input and Runtime Complexity are among the most important program properties.
  → Witnesses for non-termination indicate implementation bugs.
  → Runtime bounds indicate efficiency of a program.
• Drawback: Termination on a given input of general programs (Halting Problem) is undecidable [Turing 1937].
  → No technique which can (dis-)prove Halting Problem for all programs.
Motivation

Setting the stage

- Termination on a given input and Runtime Complexity are among the most important program properties.
  - Witnesses for non-termination indicate implementation bugs.
  - Runtime bounds indicate efficiency of a program.
- Drawback: Termination on a given input of general programs (Halting Problem) is undecidable [Turing 1937].
  - No technique which can (dis-)prove Halting Problem for all programs.
- Hope: Find sub-classes of programs where Halting Problem is decidable and runtime bounds can be computed.
Motivation

Setting the stage

- Termination on a given input and Runtime Complexity are among the most important program properties.
  - Witnesses for non-termination indicate implementation bugs.
  - Runtime bounds indicate efficiency of a program.
- Drawback: Termination on a given input of general programs (Halting Problem) is undecidable [Turing 1937].
  - No technique which can (dis-)prove Halting Problem for all programs.
- Hope: Find sub-classes of programs where Halting Problem is decidable and runtime bounds can be computed.
  - For Halting Problem: Linear loops [Li ’17, Kincaid et al. ’19]

```
while(\varphi) {
    \vec{x} \leftarrow A \cdot \vec{x}
}
```

A has entries in $\mathbb{R}_A$, $\varphi$ a linear formula.

• What about non-linear behavior in condition/update?
• What about computability of runtime bounds?
Motivation

Setting the stage

- Termination on a given input and Runtime Complexity are among the most important program properties.
  - Witnesses for non-termination indicate implementation bugs.
  - Runtime bounds indicate efficiency of a program.
- Drawback: Termination on a given input of general programs (Halting Problem) is undecidable \cite{Turing 1937}.
  - No technique which can (dis-)prove Halting Problem for all programs.
- Hope: Find sub-classes of programs where Halting Problem is decidable and runtime bounds can be computed.
  - For Halting Problem: Linear loops \cite{Li '17, Kincaid et al. '19}
    \[
    \begin{align*}
    \text{while}(\varphi) & \{
    \\
    \vec{x} & \leftarrow A \cdot \vec{x}
    \\
    \}\end{align*}
    
    A has entries in $\mathbb{R}$, $\varphi$ a linear formula.
- What about non-linear behavior in condition/update?
Motivation

Setting the stage

- Termination on a given input and Runtime Complexity are among the most important program properties.
  - Witnesses for non-termination indicate implementation bugs.
  - Runtime bounds indicate efficiency of a program.
- Drawback: Termination on a given input of general programs (Halting Problem) is undecidable [Turing 1937].
  - No technique which can (dis-)prove Halting Problem for all programs.
- Hope: Find sub-classes of programs where Halting Problem is decidable and runtime bounds can be computed.
  - For Halting Problem: Linear loops [Li ’17, Kincaid et al. ’19]
    
    ```
    \textbf{while (ϕ)} \{ \\
    \vec{x} \leftarrow A \cdot \vec{x} \\
    \}
    ```
    
    - $A$ has entries in $\mathbb{R}_A$, $\phi$ a linear formula.
- What about non-linear behavior in condition/update?
- What about computability of runtime bounds?
Motivation

Triangular Weakly Non-Linear Loops
Motivation

Triangular Weakly Non-Linear Loops

\[ \text{while}(\varphi) \{ \]
\[ \vec{x} \leftarrow \vec{a}(\vec{x}) \]
\[ \} \]
Motivation

Triangular Weakly Non-Linear Loops

while \( \varphi \) {
    \( \vec{x} \leftarrow \vec{a}(\vec{x}) \)
}

• Crucial: Computability of closed form for the iterated update.
Motivation

Triangular Weakly Non-Linear Loops

while (ϕ) { 
  \vec{x} \leftarrow a(\vec{x}) 
}

• Crucial: Computability of closed form for the iterated update.
  → Restrict ourselves to triangular weakly non-linear (twn) loops.
Motivation

Triangular Weakly Non-Linear Loops

\[
\text{while}(\varphi)\{ \\
\begin{bmatrix}
x_1 \\
\vdots \\
x_d \\
\end{bmatrix} \leftarrow \\
\begin{bmatrix}
c_1 \cdot x_1 + p_1 \\
\vdots \\
c_d \cdot x_d + p_d \\
\end{bmatrix} \\
\}
\]

- Crucial: Computability of closed form for the iterated update.

  $\rightarrow$ Restrict ourselves to triangular weakly non-linear (twn) loops.
Motivation

Triangular Weakly Non-Linear Loops

while (ϕ) {
    $\begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
    \end{bmatrix} \leftarrow 
    \begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
    \end{bmatrix}
}$
}

Crucial: Computability of closed form for the iterated update.

Variable value depends at most linearly on its previous value.

Variable value depends polynomially on previous values of variables with higher index.

Coefficients from ring $\mathbb{Z} \leq S \leq R_A$.

Variables range over $S_d$.
Motivation

Triangular Weakly Non-Linear Loops

while (ϕ) {
    \[
    \begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
    \end{bmatrix}
    \leftarrow
    \begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
    \end{bmatrix}
    \]
}

ϕ built from ∧, ∨, (¬) and polynomial inequations over S

\[
\begin{align*}
\text{while } (\varphi) \{ & \quad \varphi \text{ built from } \land, \lor, (\neg) \text{ and polynomial inequations over } S \\
\begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
    \end{bmatrix}
    \leftarrow
    \begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
    \end{bmatrix}
    \}
\end{align*}
\]
Motivation

Triangular Weakly Non-Linear Loops

\[
\begin{align*}
\text{while}(\varphi)\{ & \\
\begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
\end{bmatrix} & \leftarrow \\
\begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
\end{bmatrix}
\}
\end{align*}
\]

\(\varphi\) built from \(\land, \lor, (\neg)\) and polynomial inequations over \(S\)

\(c_i \in S\)
Motivation

Triangular Weakly Non-Linear Loops

while (ϕ) {
    \[
    \begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
    \end{bmatrix}
    \leftarrow
    \begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
    \end{bmatrix}
    \]
}

- Variable value depends at most linearly on its previous value.
Motivation

Triangular Weakly Non-Linear Loops

\[
\text{while}(\varphi)\{
\begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
\end{bmatrix}
\leftarrow
\begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
\end{bmatrix}
\}
\]

- Variable value depends at most \textit{linearly} on its previous value.
  \rightarrow Prohibits super-exponential growth as in \( x_1 \leftarrow x_1^2 \).

\varphi \text{ built from } \land, \lor, (\neg) \text{ and polynomial inequations over } S
\]
\begin{align*}
c_i & \in S \\
Z \leq S & \leq R
\end{align*}
Motivation

Triangular Weakly Non-Linear Loops

while (ϕ) { 
  $\begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \vdots \\ c_d \cdot x_d + p_d \end{bmatrix}$
}

- Variable value depends at most linearly on its previous value.
  → Prohibits super-exponential growth as in $x_1 \leftarrow x_1^2$.

ϕ built from $\land$, $\lor$, $(\neg)$ and polynomial inequations over $S$
$c_i \in S$, $p_i \in S[x_{i+1}, \ldots, x_d]$

LPAR-23 – Polynomial Loops: Beyond Termination – Hark, Frohn, Giesl – January 13 – Slide 3
Motivation

Triangular Weakly Non-Linear Loops

while (ϕ) { ϕ built from ∧, ∨, (¬) and polynomial inequations over $S$

$c_i \in S$, $p_i \in S[x_{i+1}, \ldots, x_d]$

} $\varphi$ built from $\land$, $\lor$, ($\neg$) and polynomial inequations over $S$

- Variable value depends at most linearly on its previous value.
  → Prohibits super-exponential growth as in $x_1 \leftarrow x_1^2$.
- Variable value depends polynomially on previous values of variables with higher index.

LPAR-23 – Polynomial Loops: Beyond Termination – Hark, Frohn, Giesl – January 13 – Slide 3
Motivation

Triangular Weakly Non-Linear Loops

\[ \text{while}(\varphi) \{ \]
\[
\begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d 
\end{bmatrix} \leftarrow \begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d 
\end{bmatrix}
\]
\[
\} \]

- Variable value depends at most \textit{linearly} on its previous value.
  \rightarrow \text{Prohibits super-exponential growth as in } x_1 \leftarrow x_1^2.
- Variable value depends \textit{polynomially} on previous values of variables with higher index.
  \rightarrow \text{Prohibits circular dependencies.}

\varphi \text{ built from } \land, \lor, (\neg) \text{ and polynomial inequations over } S
\[ c_i \in S, \ p_i \in S[x_{i+1}, \ldots, x_d] \]
Motivation

Triangular Weakly Non-Linear Loops

\[
\text{while}(\varphi) \{
\begin{bmatrix}
x_1 \\
\vdots \\
x_d 
\end{bmatrix} \leftarrow 
\begin{bmatrix}
c_1 \cdot x_1 + p_1 \\
\vdots \\
c_d \cdot x_d + p_d 
\end{bmatrix}
\}
\]

- \(\varphi\) built from \(\land, \lor, (\neg)\) and polynomial inequations over \(S\)
- \(c_i \in S, p_i \in S[x_{i+1}, \ldots, x_d]\)
- \(Z \leq S \leq \mathbb{R}_A\)

- Variable value depends at most \textit{linearly} on its previous value.
  \(\rightarrow\) Prohibits super-exponential growth as in \(x_1 \leftarrow x_1^2\).
- Variable value depends \textit{polynomially} on previous values of variables with higher index.
  \(\rightarrow\) Prohibits circular dependencies.
Motivation

Triangular Weakly Non-Linear Loops

\[
\text{while } (\varphi) \{
\begin{bmatrix}
 x_1 \\
 \vdots \\
 x_d
\end{bmatrix}
\leftarrow
\begin{bmatrix}
 c_1 \cdot x_1 + p_1 \\
 \vdots \\
 c_d \cdot x_d + p_d
\end{bmatrix}
\}
\]

\[\varphi \text{ built from } \wedge, \lor, (\neg) \text{ and polynomial inequations over } S\]

- Variable value depends at most \textbf{linearly} on its previous value.
  \[\rightarrow\] Prohibits super-exponential growth as in \(x_1 \leftarrow x_1^2\).

- Variable value depends \textbf{polynomially} on previous values of variables with higher index.
  \[\rightarrow\] Prohibits circular dependencies.

- Coefficients from ring \(\mathbb{Z} \leq S \leq \mathbb{R}_A\).
Motivation

Triangular Weakly Non-Linear Loops

\[ \text{while } (\varphi) \{ \begin{array}{c|c}
    x_1 & c_1 \cdot x_1 + p_1 \\
    \vdots & \vdots \\
    x_d & c_d \cdot x_d + p_d \\
  \end{array} \} \]

- \( \varphi \) built from \( \wedge, \vee, (\neg) \) and polynomial inequations over \( S \)
- \( c_i \in S, p_i \in S[x_{i+1}, \ldots, x_d] \)
- \( \mathbb{Z} \leq S \leq \mathbb{R}_A \)

- Variable value depends at most \textit{linearly} on its previous value.
  \( \rightarrow \) Prohibits super-exponential growth as in \( x_1 \leftarrow x_1^2 \).
- Variable value depends \textit{polynomially} on previous values of variables with higher index.
  \( \rightarrow \) Prohibits circular dependencies.
- Coefficients from ring \( \mathbb{Z} \leq S \leq \mathbb{R}_A \).
- Variables range over \( S^d \).
Motivation

Triangular Weakly Non-Linear Loops

\[
\text{while}( \varphi ) \{ \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \vdots \\ c_d \cdot x_d + p_d \end{bmatrix} \}
\]

\( \varphi \) built from \( \land, \lor, (\neg) \) and polynomial inequations over \( S \)

\( c_i \in S, \; p_i \in S[x_{i+1}, \ldots, x_d] \)

\( Z \leq S \leq \mathbb{R}_A \)

- Variable value depends at most \textit{linearly} on its previous value.
  - \( \rightarrow \) Prohibits super-exponential growth as in \( x_1 \leftarrow x_1^2 \).
- Variable value depends \textit{polynomially} on previous values of variables with higher index.
  - \( \rightarrow \) Prohibits circular dependencies.
- Coefficients from ring \( \mathbb{Z} \leq S \leq \mathbb{R}_A \).
- Variables range over \( S^d \).

\[
\text{while}( x_1 + x_2^2 > 0 ) \{ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} 1 \cdot x_1 + x_2^2 \cdot x_3 \\ 1 \cdot x_2^2 - 2 \cdot x_3^2 \\ 1 \cdot x_3 \end{bmatrix} \}
\]
Motivation

Contribution

while (φ) {

\[ \begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
\end{bmatrix} \leftarrow \begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
\end{bmatrix} \]

φ built from ∧, ∨, (¬) and polynomial inequations over \( S \)

\[ c_i \in S, \ p_i \in S[x_{i+1}, \ldots, x_d] \]

\[ \mathbb{Z} \leq S \leq \mathbb{R}_A \]
Motivation

Contribution

\[ \text{while}(\varphi) \{ \begin{array}{c} x_1 \\ \vdots \\ x_d \end{array} \leftarrow \begin{array}{c} c_1 \cdot x_1 + p_1 \\ \vdots \\ c_d \cdot x_d + p_d \end{array} \} \]

\[ \varphi \text{ built from } \land, \lor, (\neg) \text{ and polynomial inequations over } S \]

\[ c_i \in S, \ p_i \in S[x_{i+1}, \ldots, x_d] \]

\[ \mathbb{Z} \leq S \leq \mathbb{R}_A \]

- Halting Problem is decidable for \( S = \mathbb{R}_A \).
Motivation

Contribution

while (\( \varphi \)) {
\[
\begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
\end{bmatrix}
\leftarrow
\begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
\end{bmatrix}
\]
\( \varphi \) built from \( \land, \lor, (\neg) \) and polynomial inequations over \( S \)
\( c_i \in S, p_i \in S[x_{i+1}, \ldots, x_d] \)
\( \mathbb{Z} \leq S \leq \mathbb{R}_A \)
}

- Halting Problem is decidable for \( S = \mathbb{R}_A \).
  \( \rightarrow \) Halting problem is decidable for \( S \in \{ \mathbb{Z}, \mathbb{Q}, \mathbb{R}_A \} \).
Motivation

Contribution

while ( \( \phi \) ) {
    \[
        \begin{bmatrix}
            x_1 \\
            \vdots \\
            x_d
        \end{bmatrix}
        \leftarrow
        \begin{bmatrix}
            c_1 \cdot x_1 + p_1 \\
            \vdots \\
            c_d \cdot x_d + p_d
        \end{bmatrix}
    \]
}

\( \phi \) built from \( \land, \lor, (\neg) \) and polynomial inequations over \( S \)

\( c_i \in S, p_i \in S[x_{i+1}, \ldots, x_d] \)

\( \mathbb{Z} \leq S \leq \mathbb{R}_A \)

- Halting Problem is decidable for \( S = \mathbb{R}_A \).
- Upper runtime bounds in size of input can always be computed if \( S = \mathbb{Z} \).
Motivation

Contribution

while ( ϕ ) {
        \[ x_1 \leftarrow c_1 \cdot x_1 + p_1 \]
        . . .
        \[ x_d \leftarrow c_d \cdot x_d + p_d \]
}  

ϕ built from ∧, ∨, (¬) and polynomial inequations over S

c_i ∈ S, p_i ∈ S[x_{i+1}, \ldots, x_d]

\[ Z \leq S \leq \mathbb{R}_A \]

• Halting Problem is decidable for \( S = \mathbb{R}_A \).

• Upper runtime bounds in size of input can always be computed if \( S = \mathbb{Z} \).
  → Concept of size unclear over \( S \in \{ \mathbb{Q}, \mathbb{R}_A \} \) (see paper).
Motivation

Outline

Motivation

Computing Closed Forms

The Halting Problem

Runtime Bounds

Conclusion
Computing Closed Forms

General Idea

while \( \varphi \) {
\[
\begin{bmatrix}
  x_1 \\
  \vdots \\
  x_d 
\end{bmatrix} \leftarrow \begin{bmatrix}
  c_1 \cdot x_1 + p_1 \\
  \vdots \\
  c_d \cdot x_d + p_d
\end{bmatrix}
\]
}\n
• Closed form \( \vec{q}(\vec{e}, n) \) of the update: Values of \( \vec{x} \) after \( n \) iterations expressed in initial values \( \vec{e} \).

• Replace variables by closed form in condition \( \rightarrow \forall n \in \mathbb{N}. \varphi[\vec{x}/\vec{q}(\vec{e}, n)] \rightarrow \forall n \in \mathbb{N}. \varphi[\vec{x}/\vec{q}(\vec{e}, n)] \iff \vec{e} \text{ witnesses non-termination.} \)

\( \rightarrow M = \min_{n \in \mathbb{N}} \neg \varphi[\vec{x}/\vec{q}(\vec{e}, n)] \leftrightarrow \text{Loop does } M \text{ iterations on } \vec{e}. \)
Computing Closed Forms

General Idea

\[
\text{while}(\varphi)\{
\begin{bmatrix}
 x_1 \\
 \vdots \\
 x_d
\end{bmatrix}
\leftarrow
\begin{bmatrix}
 c_1 \cdot x_1 + p_1 \\
 \vdots \\
 c_d \cdot x_d + p_d
\end{bmatrix}
\}
\]

- Closed form \( \vec{q}(\vec{e}, n) \) of the update: Values of \( \vec{x} \) after \( n \) iterations expressed in initial values \( \vec{e} \).
Computing Closed Forms

General Idea

while (ϕ) {
    \[
    \begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
    \end{bmatrix}
    \leftarrow
    \begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
    \end{bmatrix}
    \]
}

• Closed form \( \vec{q}(\vec{e}, n) \) of the update: Values of \( \vec{x} \) after \( n \) iterations expressed in initial values \( \vec{e} \).

• Replace variables by closed form in condition

→ \( M = \min_{n \in \mathbb{N}} \neg \varphi[\vec{x}/\vec{q}(\vec{e}, n)] \)
→ \( M = \min_{n \in \mathbb{N}} \neg \varphi[\vec{x}/\vec{q}(\vec{e}, n)] \) ⇐⇒ Loop does \( M \) iterations on \( \vec{e} \).
Computing Closed Forms

General Idea

```plaintext
while (ϕ) {
  \[
  \begin{pmatrix}
  x_1 \\
  \vdots \\
  x_d \\
  \end{pmatrix}
  \leftarrow
  \begin{pmatrix}
  c_1 \cdot x_1 + p_1 \\
  \vdots \\
  c_d \cdot x_d + p_d \\
  \end{pmatrix}
  \]
}
```

- Closed form $\tilde{q}(\vec{e}, n)$ of the update: Values of $\vec{x}$ after $n$ iterations expressed in initial values $\vec{e}$.
- Replace variables by closed form in condition

$$\rightarrow \forall n \in \mathbb{N}. \varphi[\vec{x}/\tilde{q}(\vec{e}, n)]$$
Computing Closed Forms

General Idea

while (ϕ) {
\[
\begin{bmatrix}
  x_1 \\
  \vdots \\
  x_d
\end{bmatrix}
\leftarrow
\begin{bmatrix}
  c_1 \cdot x_1 + p_1 \\
  \vdots \\
  c_d \cdot x_d + p_d
\end{bmatrix}
\]
} 

- Closed form $\vec{q}(\vec{e}, n)$ of the update: Values of $\vec{x}$ after $n$ iterations expressed in initial values $\vec{e}$.
- Replace variables by closed form in condition

$$\forall n \in \mathbb{N}. \varphi[\vec{x}/\vec{q}(\vec{e}, n)] \iff \vec{e} \text{ witnesses non-termination.}$$
Computing Closed Forms

General Idea

while (ϕ) {

\[
\begin{bmatrix}
  x_1 \\
  \vdots \\
  x_d
\end{bmatrix} \leftarrow
\begin{bmatrix}
  c_1 \cdot x_1 + p_1 \\
  \vdots \\
  c_d \cdot x_d + p_d
\end{bmatrix}
\]

}

- Closed form $\vec{q}(\vec{e}, n)$ of the update: Values of $\vec{x}$ after $n$ iterations expressed in initial values $\vec{e}$.
- Replace variables by closed form in condition

\[
\forall n \in \mathbb{N}. \ \varphi[\vec{x}/\vec{q}(\vec{e}, n)] \iff \vec{e} \text{ witnesses non-termination.}
\]

\[
M = \min_{n \in \mathbb{N}} \neg \varphi[\vec{x}/\vec{q}(\vec{e}, n)]
\]
Computing Closed Forms

**General Idea**

while (ϕ) {

\[
\begin{bmatrix}
    x_1 \\
    \cdot \\
    x_d
\end{bmatrix} \leftarrow \begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \cdot \\
    c_d \cdot x_d + p_d
\end{bmatrix}
\]

}\n
- Closed form $\vec{q}(\vec{e}, n)$ of the update: Values of $\vec{x}$ after $n$ iterations expressed in initial values $\vec{e}$.
- Replace variables by closed form in condition

\[
\forall n \in \mathbb{N}. \ \varphi[\vec{x}/\vec{q}(\vec{e}, n)] \iff \vec{e} \text{ witnesses non-termination}.
\]

\[
\Rightarrow M = \min_{n \in \mathbb{N}} \neg \varphi[\vec{x}/\vec{q}(\vec{e}, n)] \iff \text{Loop does } M \text{ iterations on } \vec{e}.
\]
Computing Closed Forms

Poly-Exponential Expressions

while ($\varphi$) {
  \[
  \begin{bmatrix}
  x_1 \\
  \vdots \\
  x_d
  \end{bmatrix}
  \leftarrow
  \begin{bmatrix}
  c_1 \cdot x_1 + p_1 \\
  \vdots \\
  c_d \cdot x_d + p_d
  \end{bmatrix}
  \]
}

• Closed forms for twin-loops: Poly-Exponential Expressions.
• Closed form is computable.
• $x_4 \leftarrow 2 \cdot x_4$ has closed form $2^n \cdot e^4$. 
Computing Closed Forms

Poly-Exponential Expressions

while (\( \varphi \)) {
  \[
  \begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
  \end{bmatrix}
  \leftarrow
  \begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
  \end{bmatrix}
  \]
}

- Closed forms for twn-loops: Poly-Exponential Expressions.
Computing Closed Forms

Poly-Exponential Expressions

\[
\text{while}(\varphi)\{
\begin{bmatrix}
    x_1 \\
    \vdots \\
    x_d
\end{bmatrix}
\leftarrow
\begin{bmatrix}
    c_1 \cdot x_1 + p_1 \\
    \vdots \\
    c_d \cdot x_d + p_d
\end{bmatrix}
\}
\]

- Closed forms for twm-loops: Poly-Exponential Expressions.
- Closed form is computable.
Computing Closed Forms

Poly-Exponential Expressions

while \((x_1 + x_2^2 > 0)\) {
\[
\begin{bmatrix}
    x_1 \\
    x_2 \\
    x_3
\end{bmatrix}
\leftarrow
\begin{bmatrix}
    x_1 + x_2^2 \cdot x_3 \\
    x_2 - 2 \cdot x_3^2 \\
    x_3
\end{bmatrix}
\]
}

- Closed forms for twn-loops: Poly-Exponential Expressions.
- Closed form is computable.
Computing Closed Forms

Poly-Exponential Expressions

while \((x_1 + x_2^2 > 0)\) {
\[
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\leftarrow
\begin{bmatrix}
x_1 + x_2^2 \cdot x_3 \\
x_2 - 2 \cdot x_3^2 \\
x_3
\end{bmatrix}
\]
}

- Closed forms for twm-loops: Poly-Exponential Expressions.
- Closed form is computable.
Computing Closed Forms

Poly-Exponential Expressions

while \((x_1 + x_2^2 > 0)\) {
\[
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3
\end{bmatrix}
\leftarrow
\begin{bmatrix}
  x_1 + x_2^2 \cdot x_3 \\
  x_2 - 2 \cdot x_3^2 \\
  x_3
\end{bmatrix}
\]
}

- Closed forms for two-loops: Poly-Exponential Expressions.
- Closed form is computable.
- \(x_4 \leftarrow 2 \cdot x_4\) has closed form \(2^n \cdot e_4\).
Computing Closed Forms

Poly-Exponential Expressions

\[
\text{while } (x_1 + x_2^2 > 0) \{
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} \leftarrow \begin{bmatrix}
x_1 + x_2^2 \cdot x_3 \\
x_2 - 2 \cdot x_3^2 \\
x_3
\end{bmatrix}
\}
\]

\[
\begin{bmatrix}
e_1 + \frac{4}{3} \cdot e_3^5 \cdot n^3 + ( -2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3 ) \cdot n^2 + ( e_2^2 \cdot e_3 + \frac{2}{3} \cdot e_3^5 + 2 \cdot e_2 \cdot e_3^3 ) \cdot n \\
e_2 - 2 \cdot e_3^2 \cdot n \\
e_3
\end{bmatrix}
\]

Truth-value of condition after \( n \) iterations on initial values \( \vec{e} \).
Computing Closed Forms

Poly-Exponential Expressions

\[
\text{while}(x_1 + x_2^2 > 0) \{
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} \leftarrow \begin{bmatrix}
x_1 + x_2^2 \cdot x_3 \\
x_2 - 2 \cdot x_3^2 \\
x_3
\end{bmatrix}
\}
\[
\begin{bmatrix}
e_1 + 4/3 \cdot e_3^5 \cdot n^3 + (-2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3) \cdot n^2 + (e_2^2 \cdot e_3 + 2/3 \cdot e_3^3 + 2 \cdot e_2 \cdot e_3^3) \cdot n \\
e_2 - 2 \cdot e_3^2 \cdot n \\
e_3
\end{bmatrix}
\]
Computing Closed Forms

Poly-Exponential Expressions

while \((x_1 + x_2^2 > 0)\) {
\[
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3
\end{bmatrix} \leftarrow \begin{bmatrix}
  x_1 + x_2^2 \cdot x_3 \\
  x_2 - 2 \cdot x_3^2 \\
  x_3
\end{bmatrix}
\]
}\]

\[
\begin{bmatrix}
  e_1 + \frac{4}{3} \cdot e_3^5 \cdot n^3 + (-2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3) \cdot n^2 + (e_2^2 \cdot e_3 + \frac{2}{3} \cdot e_3^2 + 2 \cdot e_2 \cdot e_3^3) \cdot n + (e_1 + e_2^2) \\
  e_2 - 2 \cdot e_3^2 \cdot n \\
  e_3
\end{bmatrix}
\]

Truth-value of condition after \(n\) iterations on initial values \(\vec{e}\).
Computing Closed Forms

Poly-Exponential Expressions

\[
\text{while } (x_1 + x_2^2 > 0) \{ \\
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} x_1 + x_2^2 \cdot x_3 \\ x_2 - 2 \cdot x_2^3 \\ x_3 \end{bmatrix} \\
\}
\]

\[
\begin{bmatrix}
e_1 + 4 \cdot e_3^5 \cdot n^3 + (-2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3) \cdot n^2 + (e_2^2 \cdot e_3 + \frac{2}{3} \cdot e_3^5 + 2 \cdot e_2 \cdot e_3^3) \cdot n \\
e_2 - 2 \cdot e_2^2 \cdot n \\
e_3
\end{bmatrix}
\]

\[
(\frac{4}{3} \cdot e_3^5) \cdot n^3 + (-2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3 + 4 \cdot e_3^4) \cdot n^2 \\
+ (e_2^2 \cdot e_3 + \frac{2}{3} \cdot e_3^5 + 2 \cdot e_2 \cdot e_3^3 - 4 \cdot e_2 \cdot e_3^2) \cdot n + (e_1 + e_2^2)
\]

→ Truth-value of condition after \(n\) iterations on initial values \(\vec{e}\).
The Halting Problem

Dominant term

\[
\left(\frac{4}{3} \cdot e_3^5\right) \cdot n^3 + \left(-2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3 + 4 \cdot e_3^4\right) \cdot n^2 \\
+ \left(e_2^2 \cdot e_3 + \frac{2}{3} \cdot e_3^5 + 2 \cdot e_2 \cdot e_3^3 - 4 \cdot e_2 \cdot e_3^2\right) \cdot n + \left(e_1 + e_2^2\right)
\]
The Halting Problem

Dominant term

\[
\left(\frac{4}{3}e_3^5\right) \cdot n^3 + \left(-2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3 + 4 \cdot e_3^4\right) \cdot n^2 \\
+ \left(e_2^2 \cdot e_3 + \frac{2}{3}e_3^5 + 2 \cdot e_2 \cdot e_3^3 - 4 \cdot e_2 \cdot e_3^2\right) \cdot n + (e_1 + e_2^2)
\]

- Decide halting problem for \( \vec{e} = (-4, 2, 1) \).
The Halting Problem

Dominant term

\[
\frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3}
\]

- Decide halting problem for \( \vec{e} = (-4, 2, 1) \).

\[
\frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} > 0
\]

→ Upper bound on \( N \) is always computable (see paper).
→ Here \( N \leq 4 \).
\[\Rightarrow \forall n \geq 4. \quad \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} > 0\]
The Halting Problem

Dominant term

\[ \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} \]

- Decide halting problem for \( \vec{e} = (-4, 2, 1) \).
- \( \frac{4}{3} \cdot n^3 \) is dominant and positive.
The Halting Problem

Dominant term

\[ \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} \]

- Decide halting problem for \( \vec{e} = (-4, 2, 1) \).
- \( \frac{4}{3} \cdot n^3 \) is dominant and positive.
  \[ \Rightarrow \text{Expression is eventually positive } (n \to \infty). \]
The Halting Problem

Dominant term

\[ \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} \]

- Decide halting problem for \( \vec{e} = (-4, 2, 1) \).
- \( \frac{4}{3} \cdot n^3 \) is dominant and positive.
  - Expression is eventually positive (\( n \to \infty \)).
  - From certain \( N \) onwards dominant term dominates remaining expression.
**The Halting Problem**

**Dominant term**

\[
\frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3}
\]

- Decide halting problem for \( \vec{e} = (-4, 2, 1) \).
- \( \frac{4}{3} \cdot n^3 \) is dominant and positive.
  - Expression is eventually positive \( (n \to \infty) \).
  - From certain \( N \) onwards dominant term dominates remaining expression.
  - Upper bound on \( N \) is always computable (see paper).
The Halting Problem

Dominant term

\[
\frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3}
\]

- Decide halting problem for \( \vec{e} = (-4, 2, 1) \).
- \( \frac{4}{3} \cdot n^3 \) is dominant and positive.
  - Expression is eventually positive (\( n \to \infty \)).
  - From certain \( N \) onwards dominant term dominates remaining expression.
  - Upper bound on \( N \) is always computable (see paper).
  - Here \( N \leq 4 \).
The Halting Problem

Dominant term

\[ \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} \]

- Decide halting problem for \( \vec{e} = (-4, 2, 1) \).
- \( \frac{4}{3} \cdot n^3 \) is dominant and positive.
  - Expression is eventually positive (\( n \to \infty \)).
  - From certain \( N \) onwards dominant term dominates remaining expression.
  - Upper bound on \( N \) is always computable (see paper).
  - Here \( N \leq 4 \).

\[ \Rightarrow \forall n \geq 4. \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} > 0 \]
Deciding the Halting Problem

\forall n \geq 4. \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} > 0
Deciding the Halting Problem

\[ \forall n \geq 4. \ 4 \cdot \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} > 0 \]

- Truth value for \( n \geq 4 \) is known.

\[ \Rightarrow \text{Only finitely many cases remain.} \]

- Here: \( n = 0, 1, 2, ... \)
The Halting Problem

Deciding the Halting Problem

\[ \forall n \geq 4. \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} > 0 \]

• Truth value for \( n \geq 4 \) is known.
  → Only finitely many cases remain.
The Halting Problem

Deciding the Halting Problem

\[ \forall n \geq 4. \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} > 0 \]

- Truth value for \( n \geq 4 \) is known.
  - Only finitely many cases remain.
  - Here: \( n = 0, 1, 2, 3 \).
Deciding the Halting Problem

\[ \forall n \geq 4. \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} > 0 \]

- Truth value for \( n \geq 4 \) is known.
  → Only finitely many cases remain.
  → Here: \( n = 0, 1, 2, 3 \).

\[
\begin{align*}
\frac{4}{3} \cdot 0^3 - 2 \cdot 0^2 + \frac{2}{3} &> 0 \\
\frac{4}{3} \cdot 1^3 - 2 \cdot 1^2 + \frac{2}{3} &< 0 \\
\frac{4}{3} \cdot 2^3 - 2 \cdot 2^2 + \frac{2}{3} &> 0 \\
\frac{4}{3} \cdot 3^3 - 2 \cdot 3^2 + \frac{2}{3} &> 0
\end{align*}
\]
The Halting Problem

Deciding the Halting Problem

∀n \geq 4. \frac{4}{3} \cdot n^3 - 2 \cdot n^2 + \frac{2}{3} > 0

• Truth value for n \geq 4 is known.
  → Only finitely many cases remain.
  → Here: n = 0, 1, 2, 3.

\[
\begin{align*}
\frac{4}{3} \cdot 0^3 - 2 \cdot 0^2 + \frac{2}{3} &> 0 \\
\frac{4}{3} \cdot 1^3 - 2 \cdot 1^2 + \frac{2}{3} &< 0 \\
\frac{4}{3} \cdot 2^3 - 2 \cdot 2^2 + \frac{2}{3} &> 0 \\
\frac{4}{3} \cdot 3^3 - 2 \cdot 3^2 + \frac{2}{3} &> 0
\end{align*}
\]

→ Loop halts on \bar{e} = (-4, 2, 1).
Deciding the Halting Problem

- Truth value for $n \geq N$ is known.
  - Only finitely many cases remain.
The Halting Problem

Deciding the Halting Problem

• Truth value for $n \geq N$ is known.
  → Only finitely many cases remain.

→ The Halting Problem for twin-loops is decidable.
The Halting Problem

Deciding the Halting Problem

- Truth value for $n \geq N$ is known.
  - Only finitely many cases remain.

  → The Halting Problem for twin-loops is decidable.

  → Witnesses for Non-Termination are enumerable.
Runtime Bounds

Outline

Motivation

Computing Closed Forms

The Halting Problem

Runtime Bounds

Conclusion
Runtime Bounds

Reminder

while \((x_1 + x_2)^2 > 0\) {
  \[
  \begin{pmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  \end{pmatrix}
  \leftarrow
  \begin{pmatrix}
  x_1 + x_2^2 \cdot x_3 \\
  x_2 - 2 \cdot x_2^3 \\
  x_3 \\
  \end{pmatrix}
  \]
}

\[
\begin{pmatrix}
  e_1 + 4^3 \ldots \text{condition after } n \text{ iterations on initial values } \vec{e}.
\end{pmatrix}
\]

→ Termination reached whenever expression \(\leq 0\).

→ Rescale for simplicity.
Runtime Bounds

Reminder

while \((x_1 + x_2^2 > 0)\) {
    
    \[
    \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} x_1 + x_2^2 \cdot x_3 \\ x_2 - 2 \cdot x_2^2 \\ x_3 \end{bmatrix}
    \]

    
    \[
    \begin{bmatrix} e_1 + \frac{4}{3} \cdot e_3^5 \cdot n^3 + \left( -2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3 + 4 \cdot e_3^4 \right) \cdot n^2 + \left( e_2^2 \cdot e_3 + \frac{2}{3} \cdot e_3^5 + 2 \cdot e_2 \cdot e_3^3 - 4 \cdot e_2 \cdot e_3^2 \right) \cdot n + \left( e_1 + e_2^2 \right) e_3 - 2 \cdot e_3^2 \cdot n 
    \end{bmatrix}
    \]
}

Truth-value of condition after \(n\) iterations on initial values \(\vec{e}\).

Termination reached whenever expression \(\leq 0\).

Rescale for simplicity.
Runtime Bounds

Reminder

while \((x_1 + x_2^2 > 0)\) {
\[
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3
\end{bmatrix}
\leftarrow
\begin{bmatrix}
  x_1 + x_2^2 \cdot x_3 \\
  x_2 - 2 \cdot x_3^2 \\
  x_3
\end{bmatrix}
\]
}
\[
\begin{bmatrix}
  e_1 + \frac{4}{3} \cdot e_3^5 \cdot n^3 + (-2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3) \cdot n^2 + (e_2^2 \cdot e_3 + \frac{2}{3} \cdot e_3^5 + 2 \cdot e_2 \cdot e_3^3) \cdot n \\
  e_2 - 2 \cdot e_3^2 \cdot n \\
  e_3
\end{bmatrix}
\]

\[\rightarrow\] Truth-value of condition after \(n\) iterations on initial values \(\vec{e}\).
Runtime Bounds

Reminder

while \( (x_1 + x_2^2 > 0) \) {
\[
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3
\end{bmatrix} \leftarrow \begin{bmatrix}
  x_1 + x_2^2 \cdot x_3 \\
  x_2 - 2 \cdot x_2^3 \\
  x_3
\end{bmatrix}
\]
\}

\[
\begin{bmatrix}
  e_1 + \frac{4}{3} \cdot e_3^5 \cdot n^3 + \left( -2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3 + 4 \cdot e_4^3 \right) \cdot n^2 + \left( e_2^2 \cdot e_3 + \frac{2}{3} \cdot e_3^5 + 2 \cdot e_2 \cdot e_3^3 - 4 \cdot e_2 \cdot e_3^2 \right) \cdot n + \left( e_1 + e_2^2 \right)
\end{bmatrix}
\]

→ Truth-value of condition after \( n \) iterations on initial values \( \vec{e} \).
→ Termination reached whenever expression \( \leq 0 \).
while \((x_1 + x_2^2 > 0)\) {
    \[
    \begin{bmatrix}
        x_1 \\
        x_2 \\
        x_3
    \end{bmatrix}
    \leftarrow
    \begin{bmatrix}
        x_1 + x_2^2 \cdot x_3 \\
        x_2 - 2 \cdot x_2^2 \\
        x_3
    \end{bmatrix}
    \]
}\}

\[
\begin{bmatrix}
    e_1 + \frac{4}{3} \cdot e_3^5 \cdot n^3 + (-2 \cdot e_3^5 - 2 \cdot e_2 \cdot e_3^3) \cdot n^2 + (e_2^2 \cdot e_3 + \frac{2}{3} \cdot e_3^5 + 2 \cdot e_2 \cdot e_3^3) \cdot n \\
    e_2 - 2 \cdot e_3^2 \cdot n \\
    e_3
\end{bmatrix}
\]

\[
(4 \cdot e_3^5) \cdot n^3 + (-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4) \cdot n^2 \\
+ (3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^3) \cdot n + 3 \cdot (e_1 + e_2^2)
\]

→ Truth-value of condition after \(n\) iterations on initial values \(\vec{e}\).
→ Termination reached whenever expression \(\leq 0\).
→ Rescale for simplicity.
Dominant Term

\[
(4 \cdot e_3^5) \cdot n^3 + (-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4) \cdot n^2 \\
+ (3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2) \cdot n + 3 \cdot (e_1 + e_2^2)
\]
Dominant Term

\[(4 \cdot e_3^5) \cdot n^3 + (-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4) \cdot n^2 + (3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2) \cdot n + 3 \cdot (e_1 + e_2^2)\]

- If \(e_3 \neq 0 \implies (4 \cdot e_3^5) \cdot n^3\) is dominant.
Runtime Bounds

Dominant Term

\[
(4 \cdot e_3^5) \cdot n^3 + (-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^2 + 12 \cdot e_3^4) \cdot n^2 \\
+ (3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2) \cdot n + 3 \cdot (e_1 + e_2^2)
\]

• If \( e_3 \neq 0 \) \( \implies \) \( (4 \cdot e_3^5) \cdot n^3 \) is dominant.

\[\rightarrow\] From certain \( N(\vec{e}) \) onwards dominant term dominates remaining expression.
Runtime Bounds

Dominant Term

\[
(4 \cdot e_3^5) \cdot n^3 + (-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4) \cdot n^2 \\
+ (3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2) \cdot n + 3 \cdot (e_1 + e_2^2)
\]

- If \( e_3 \neq 0 \) \( \Rightarrow \) \( (4 \cdot e_3^5) \cdot n^3 \) is dominant.

\[\rightarrow\] From certain \( N(\vec{e}) \) onwards dominant term dominates remaining expression.

\[\rightarrow\] Upper bound on \( N(\vec{e}) \) depending only on \( \|\vec{e}\| \) for arbitrary \( \vec{e} \) is always computable.
Runtime Bounds

Dominant Term

\((4 \cdot e_3^5) \cdot n^3 + (-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4) \cdot n^2
+ (3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2) \cdot n + 3 \cdot (e_1 + e_2^2)\)

• If \(e_3 \neq 0 \implies (4 \cdot e_3^5) \cdot n^3\) is dominant.

→ From certain \(N(\vec{e})\) onwards dominant term dominates remaining expression.
→ Upper bound on \(N(\vec{e})\) depending only on \(\|\vec{e}\|\) for arbitrary \(\vec{e}\) is always computable.
→ Yields upper bound on runtime.
Runtime Bounds

Upper Bound

\[
(4 \cdot e_3^5) \cdot n^3 + (-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4) \cdot n^2 \\
+ (3 \cdot e_2^2 - e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2) \cdot n + 3 \cdot (e_1 + e_2^2)
\]
Runtime Bounds

Upper Bound

\[
(4 \cdot e_3^5) \cdot n^3 + (-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4) \cdot n^2 \\
+ (3 \cdot e_2^5 \cdot e_3 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2) \cdot n + 3 \cdot (e_1 + e_2^2)
\]

- Idea: compute bound such that \( n^3 \) dominates remaining expression.
Upper Bound

\[ n^3 > \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_4^3 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^4 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot \left( e_1 + e_2^2 \right) \]

• Idea: compute bound such that \( n^3 \) dominates remaining expression.
Runtime Bounds

Upper Bound

\[ n^3 > \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_4^3 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot (e_1 + e_2^2) \]

- Idea: compute bound such that \( n^3 \) dominates remaining expression.
  - \( \rightarrow \) Compute upper bound on all coefficients first.
Runtime Bounds

Upper Bound

\[ n^3 > \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot (e_1 + e_2^2) \]

• Idea: compute bound such that \( n^3 \) dominates remaining expression.

→ Compute upper bound on all coefficients first.

\[ -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4 \leq 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \]

\[ 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \leq 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \]

\[ 3 \cdot (e_1 + e_2^2) \leq 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \]
Runtime Bounds

Upper Bound

\[ n^3 > \left( 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \right) \cdot (n^2 + n + 1) \]

- Idea: compute bound such that \( n^3 \) dominates remaining expression.

\[ \rightarrow \text{Compute upper bound on all coefficients first.} \]

\[ -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4 \leq 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \]

\[ 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \leq 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \]

\[ 3 \cdot (e_1 + e_2^2) \leq 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \]
Runtime Bounds

Upper Bound

\[ n^3 > \left( 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \right) \cdot (n^2 + n + 1) \]

- Idea: compute bound such that \( n^3 \) dominates remaining expression.
Runtime Bounds

Upper Bound

\[ n^3 > \left( 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \right) \cdot (n^2 + n + 1) \]

- Idea: compute bound such that \( n^3 \) dominates remaining expression.
  → Combine the terms of lower order.
Runtime Bounds

Upper Bound

\[ n^3 > \left(12 \cdot \sum_{j=0}^{5}(|e_1| + |e_2| + |e_3|)^j \right) \cdot (n^2 + n + 1) \]

- Idea: compute bound such that \( n^3 \) dominates remaining expression.
  → Combine the terms of lower order.
  → For \( n > 1 \) we have \( n^2 > n + 1 \)
Runtime Bounds

Upper Bound

\[ n^3 > \left( 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \right) \cdot (n^2 + n) \]

- Idea: compute bound such that \( n^3 \) dominates remaining expression.
  - Combine the terms of lower order.
  - For \( n > 1 \) we have \( n^2 > n + 1 \)
Runtime Bounds

Upper Bound

\[ n^3 > \left( 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \right) \cdot 2 \cdot n^2 \]

- Idea: compute bound such that \( n^3 \) dominates remaining expression.
  - Combine the terms of lower order.
  - For \( n > 1 \) we have \( n^2 > n + 1 \)
Runtime Bounds

Upper Bound

\[ n > \left( 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \right) \cdot 2 \]

- Idea: compute bound such that \( n^3 \) dominates remaining expression.
  - Combine the terms of lower order.
  - For \( n > 1 \) we have \( n^2 > n + 1 \)
Upper Bound

At this point, the dominant term dominates the remaining expression.

→ At this point, the loop has terminated on \( \vec{e} \) or never will.
Runtime Bounds

Upper Bound

\[ n^3 > \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot (e_1 + e_2^2) \]

\[ n > \left( 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \right) \cdot 2 \]
Runtime Bounds

Upper Bound

\[ n^3 > \left(-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4\right) \cdot n^2 + \left(3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2\right) \cdot n + 3 \cdot (e_1 + e_2^2) \]

\[ n > \left(12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j\right) \cdot 2 \]

\[ (4 \cdot e_3^5) \cdot n^3 + \left(-6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4\right) \cdot n^2 + \left(3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2\right) \cdot n + 3 \cdot (e_1 + e_2^2) \]
Runtime Bounds

Upper Bound

\[ n^3 > \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_4^3 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot (e_1 + e_2^2) \]

\[ n > \left( 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \right) \cdot 2 \]

\[ (4 \cdot e_3^5) \cdot n^3 + \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_4^3 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot (e_1 + e_2^2) \]

→ At this point, the dominant term dominates the remaining expression.
Runtime Bounds

Upper Bound

\[ n^3 > \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot (e_1 + e_2^2) \]

\[ n > \left( 12 \cdot \sum_{j=0}^{5} (|e_1| + |e_2| + |e_3|)^j \right) \cdot 2 \]

\[ (4 \cdot e_3^5) \cdot n^3 + \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot (e_1 + e_2^2) \]

→ At this point, the dominant term dominates the remaining expression.
→ At this point, the loop has terminated on \( \vec{e} \) or never will.
Upper Bound

\[ n^3 > \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot (e_1 + e_2) \]

→ At this point, the dominant term dominates the remaining expression.

\[ n > 12 \cdot \left( \sum_{j=0}^{5} \| \vec{e}^j \| \right) \cdot 2 \]

\[ (4 \cdot e_3^5) \cdot n^3 + \left( -6 \cdot e_3^5 - 6 \cdot e_2 \cdot e_3^3 + 12 \cdot e_3^4 \right) \cdot n^2 + \left( 3 \cdot e_2^2 \cdot e_3 + 2 \cdot e_3^5 + 6 \cdot e_2 \cdot e_3^3 - 12 \cdot e_2 \cdot e_3^2 \right) \cdot n + 3 \cdot (e_1 + e_2) \]

→ At this point, the loop has terminated on \( \vec{e} \) or never will.
Upper Bound

while \((x_1 + x_2)^2 > 0\) {
  \[
  \begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 
  \end{bmatrix}
  \leftarrow
  \begin{bmatrix}
  x_1 + x_2 \\
  x_2 - 2 \cdot x_2 \\
  x_3 
  \end{bmatrix}
  \]
}

If loop terminates, termination within \(12 \cdot \sum_{j=0}^{5} \|\vec{e}\|_j \cdot 2\) iterations.

\(\rightarrow\) Polynomial runtime bound for twn-loop over integers.

\(\rightarrow\) Linear twn-loops over integers have at most linear runtime (see paper).
### Upper Bound

```latex
\text{while} \ (x_1 + x_2^2 > 0) \{
\begin{bmatrix}
    x_1 \\
    x_2 \\
    x_3
\end{bmatrix} \leftarrow \begin{bmatrix}
    x_1 + x_2^2 \cdot x_3 \\
    x_2 - 2 \cdot x_2^3 \\
    x_3
\end{bmatrix}
\}
```

If the loop terminates, the runtime is within $\left(12 \cdot \sum_{j=0}^{5} \|\vec{e}\|_j \right) \cdot 2$ iterations.

→ Polynomial runtime bound for twn-loop over integers.

→ Linear twn-loops over integers have at most linear runtime (see paper).
Upper Bound

\[
\text{while} \left( x_1 + x_2^2 > 0 \right) \{
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3
\end{bmatrix} \leftarrow \begin{bmatrix}
  x_1 + x_2^2 \cdot x_3 \\
  x_2 - 2 \cdot x_2^2 \\
  x_3
\end{bmatrix}
\}
\]

If loop terminates, termination within \( \left( 12 \cdot \sum_{j=0}^{5} ||e||^j \right) \cdot 2 \) iterations.
Runtime Bounds

Upper Bound

while \((x_1 + x_2^2 > 0)\) {
\[
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3
\end{bmatrix} \leftarrow \begin{bmatrix}
  x_1 + x_2^2 \cdot x_3 \\
  x_2 - 2 \cdot x_3^2 \\
  x_3
\end{bmatrix}
\]
}

→ Polynomial runtime bound for twn-loop over integers.

If loop terminates, termination within \(12 \cdot \sum_{j=0}^{5} ||\tilde{e}||^j \cdot 2\) iterations.
Runtime Bounds

Upper Bound

while \((x_1 + x_2^2 > 0)\) {
\[
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 
\end{bmatrix}
\leftarrow
\begin{bmatrix}
  x_1 + x_2^2 \cdot x_3 \\
  x_2 - 2 \cdot x_3^2 \\
  x_3 
\end{bmatrix}
\]
}

\(\rightarrow\) Polynomial runtime bound for twn-loop over integers.

\(\rightarrow\) Linear twn-loops over integers have at most linear runtime (see paper).
Conclusion

Summary

• Halting problem for twn-loops is decidable.
• Witnesses for non-termination are enumerable.
• Upper runtime bound for integer twn-loops is always computable.
• Can handle loops out of reach for incomplete techniques using (lexicographic) ranking functions (see paper).

Future Work
• Handling of non-twn loops.
• Investigate tightness of upper runtime bounds.
• Asymptotic bound for runtime directly from loop.
Conclusion

Summary

• Halting problem for twn-loops is decidable.
Conclusion

Summary

- Halting problem for twn-loops is decidable.
- Witnesses for non-termination are enumerable.
Conclusion

Summary

- Halting problem for twn-loops is decidable.
- Witnesses for non-termination are enumerable.
- Upper runtime bound for integer twn-loops is always computable.
Conclusion

Summary

- Halting problem for twn-loops is decidable.
- Witnesses for non-termination are enumerable.
- Upper runtime bound for integer twn-loops is always computable.
- Can handle loops out of reach for incomplete techniques using (lexicographic) ranking functions (see paper).
Conclusion

Summary

• Halting problem for twn-loops is decidable.
• Witnesses for non-termination are enumerable.
• Upper runtime bound for integer twn-loops is always computable.
• Can handle loops out of reach for incomplete techniques using (lexicographic) ranking functions (see paper).
Conclusion

Summary

• Halting problem for twn-loops is decidable.
• Witnesses for non-termination are enumerable.
• Upper runtime bound for integer twn-loops is always computable.
• Can handle loops out of reach for incomplete techniques using (lexicographic) ranking functions (see paper).

Future Work
Conclusion

Summary

- Halting problem for twn-loops is decidable.
- Witnesses for non-termination are enumerable.
- Upper runtime bound for integer twn-loops is always computable.
- Can handle loops out of reach for incomplete techniques using (lexicographic) ranking functions (see paper).

Future Work

- Handling of non-twn loops.
Conclusion

Summary

• Halting problem for twn-loops is decidable.
• Witnesses for non-termination are enumerable.
• Upper runtime bound for integer twn-loops is always computable.
• Can handle loops out of reach for incomplete techniques using (lexicographic) ranking functions (see paper).

Future Work

• Handling of non-twn loops.
• Investigate tightness of upper runtime bounds.
Conclusion

Summary

• Halting problem for twn-loops is decidable.
• Witnesses for non-termination are enumerable.
• Upper runtime bound for integer twn-loops is always computable.
• Can handle loops out of reach for incomplete techniques using (lexicographic) ranking functions (see paper).

Future Work

• Handling of non-twn loops.
• Investigate tightness of upper runtime bounds.
• Asymptotic bound for runtime directly from loop.
Conclusion

Summary
- Halting problem for twn-loops is decidable.
- Witnesses for non-termination are enumerable.
- Upper runtime bound for integer twn-loops is always computable.
- Can handle loops out of reach for incomplete techniques using (lexicographic) ranking functions (see paper).

Future Work
- Handling of non-twn loops.
- Investigate tightness of upper runtime bounds.
- Asymptotic bound for runtime directly from loop.
Conclusion

Summary
- Halting problem for twn-loops is decidable.
- Witnesses for non-termination are enumerable.
- Upper runtime bound for integer twn-loops is always computable.
- Can handle loops out of reach for incomplete techniques using (lexicographic) ranking functions (see paper).

Future Work
- Handling of non-twn loops.
- Investigate tightness of upper runtime bounds.
- Asymptotic bound for runtime directly from loop.

Thank you