Polygraphic programs

Motivations
Static analysis of programs
Static analysis of programs

**Data:** the code of a (first-order functional) program

```plaintext
type integer = Z | S of integer ;;
let rec add = function
  | Z, x -> x
  | S(x), y -> S(add(x,y)) ;;
let rec mult = function
  | x, Z -> Z
  | x, S(y) -> add(x,mult(x,y)) ;;
```
Static analysis of programs

**Data:** the code of a (first-order functional) program

```ocaml
type integer = Z | S of integer ;;
let rec add = function
    | Z, x -> x
    | S(x), y -> S(add(x,y)) ;;
let rec mult = function
    | x, Z -> Z
    | x, S(y) -> add(x,mult(x,y)) ;;
```

**Questions:** properties such as

- termination,
- computational complexity,

*i.e.* memory (spatial complexity) and time (temporal complexity) required.
Polynomial interpretations of terms
Polynomial interpretations of terms

**Idea:** TRSs interpretations can bound complexity. [Hofbauer, Lautemann; Cichon, Lescanne]
Polynomial interpretations of terms

**Idea:** TRSs interpretations can bound complexity. [Hofbauer, Lautemann; Cichon, Lescanne]

1 – **Translate** the program into a TRS $(\Sigma, R)$:

\[
\begin{align*}
\Sigma : & \quad \text{constructors } (0, s), \quad \text{functions } (+, \times), \\
R : & \quad 0 + x \to x, \quad s(x) + y \to s(x + y), \quad (x \times 0) \to 0, \quad x \times s(y) \to x + (x \times y).
\end{align*}
\]
Polynomial interpretations of terms

Idea: TRSs interpretations can bound complexity. [Hofbauer, Lautemann; Cichon, Lescanne]

1 – Translate the program into a TRS \((\Sigma, R)\):

\[\begin{align*}
\Sigma : & \text{ constructors } (0, s), \text{ functions } (+, \times), \\
R : & 0 + x \rightarrow x, \quad s(x) + y \rightarrow s(x + y), \quad (x \times 0) \rightarrow 0, \quad x \times s(y) \rightarrow x + (x \times y).
\end{align*}\]

2 – Build a map \(\cdot : \Sigma \rightarrow \mathbb{N}[X, Y, \ldots]\), such as:

\[\begin{align*}
\langle0\rangle = 1, \quad \langle s\rangle = X + 1, \quad \langle+\rangle = 2X + Y + 1, \quad \langle\times\rangle = (X + 1)(Y + 1).
\end{align*}\]
Polynomial interpretations of terms

Idea: TRSs interpretations can bound complexity. [Hofbauer, Lautemann; Cichon, Lescanne]

1 – Translate the program into a TRS \((\Sigma, R)\):

\[\Sigma: \text{ constructors } (0, s), \text{ functions } (+, \times),\]
\[R: \ 0 + x \rightarrow x, \ s(x) + y \rightarrow s(x + y), \ (x \times 0) \rightarrow 0, \ x \times s(y) \rightarrow x + (x \times y).\]

2 – Build a map \((\cdot) : \Sigma \rightarrow \mathbb{N}[X, Y, \ldots]\), such as:

\[\langle 0 \rangle = 1, \ \langle s \rangle = X + 1, \ \langle + \rangle = 2X + Y + 1, \ \langle \times \rangle = (X + 1)(Y + 1).\]

Then \((\langle u \rangle) > (\langle v \rangle)\) when \(u \rightarrow v\), plus some other conditions: \((\cdot)\) is a polynomial interpretation.
Polynomial interpretations of terms

**Idea:** TRSs interpretations can bound complexity. [Hofbauer, Lautemann; Cichon, Lescanne]

1 – **Translate** the program into a TRS $(\Sigma, R)$:

- $\Sigma$: constructors (0, s), functions (+, $\times$),
- $R$: $0 + x \rightarrow x$, $s(x) + y \rightarrow s(x + y)$, $(x \times 0) \rightarrow 0$, $x \times s(y) \rightarrow x + (x \times y)$.

2 – **Build** a map $(\cdot) : \Sigma \rightarrow \mathbb{N}[X, Y, \ldots]$, such as:

- $(0) = 1$, $(s) = X + 1$, $(+) = 2X + Y + 1$, $(\times) = (X + 1)(Y + 1)$.

Then $(\cdot u) > (\cdot v)$ when $u \rightarrow v$, plus some other conditions: $(\cdot)$ is a **polynomial interpretation**.

**Results:** [Bonfante, Cichon, Marion, Touzet]

- **Space:** if $u$ is a value, $(\cdot u)$ bounds the size of $u$.
- **Time:** if $f$ is a function, then $(\cdot f(\vec{u}))$ bounds the number of steps to reach the result.
- **The values of $(\cdot)$ on constructors give the complexity class of the functions (PTIME here).**
Limitations of polynomial interpretations
Limitations of polynomial interpretations

**Mixed bounds:** \( \| \cdot \| \) bounds space *and* time.

Example: + requires X steps, not \( 2X + Y + 1 \).

Consequences: mixed information, overestimated bounds.
Limitations of polynomial interpretations

**Mixed bounds:** \(|\cdot|\) bounds space *and* time.

Example: \(\) requires \(X\) steps, not \(2X + Y + 1\).

Consequences: mixed information, overestimated bounds.

**Programs out of range:** polynomial interpretations yield simplification orders.

Example: integer division \(s(x)/y \rightarrow s((x - y)/y)\) gives \(s(x)/s(x) \preceq s((x - s(x))/s(x))\).

Consequence: cannot give bounds for such programs.
Limitations of polynomial interpretations

**Mixed bounds:** \( |\cdot| \) bounds space *and* time.

Example: + requires \( X \) steps, not \( 2X + Y + 1 \).

Consequences: mixed information, overestimated bounds.

**Programs out of range:** polynomial interpretations yield simplification orders.

Example: integer division \( s(x)/y \to s((x-y)/y) \) gives \( s(x)/s(x) \triangleleft s((x-s(x))/s(x)) \).

Consequence: cannot give bounds for such programs.

**Algorithms out of range:** TRSs can only compute *products* of functions with one output each.

Counter-example: computing *at the same time* \((x_1,x_3,\ldots)\) and \((x_2,x_4,\ldots)\) from \((x_1,x_2,x_3,\ldots)\).

Consequence: TRSs cannot describe faithfully "divide-and-conquer" algorithms.
Limitations of polynomial interpretations

**Mixed bounds:** $\langle \cdot \rangle$ bounds space and time.

Example: $+$ requires $X$ steps, not $2X + Y + 1$.

Consequences: mixed information, overestimated bounds.

**Programs out of range:** polynomial interpretations yield simplification orders.

Example: integer division $s(x)/y \rightarrow s((x - y)/y)$ gives $s(x)/s(x) \triangleq s((x - s(x))/s(x))$.

Consequence: cannot give bounds for such programs.

**Algorithms out of range:** TRSs can only compute products of functions with one output each.

Counter-example: computing at the same time $(x_1, x_3, \ldots)$ and $(x_2, x_4, \ldots)$ from $(x_1, x_2, x_3, \ldots)$.

Consequence: TRSs cannot describe faithfully "divide-and-conquer" algorithms.

**A solution:** provide an alternative framework for programs.
Polygraphic programs
Examples of polygraphic programs
Examples of polygraphic programs

Addition and multiplication:

 Computes addition $\downarrow$ and multiplication $\triangleleft$ over natural numbers $\langle \varnothing, \varnothing \rangle$. 
Examples of polygraphic programs

Addition and multiplication:

\[ \begin{align*}
\begin{array}{c}
\text{Computes addition } \triangledown \text{ and multiplication } \blacktriangle \text{ over natural numbers } \langle \diamondsuit, \heartsuit \rangle.
\end{array}
\end{align*} \]

Structure operations permutation \[\begin{array}{c}
\text{permutation } \times
\end{array}\], duplication \[\text{duplication } \triangle
\] and erasure \[\text{erasure } \circ
\] are explicit:

\[ \begin{align*}
\begin{array}{c}
\text{Structure operations}
\end{array}
\end{align*} \]
Examples of polygraphic programs

Addition and multiplication:

Computes addition \( \bigtriangledown \) and multiplication \( \bigtriangleup \) over natural numbers \( \langle \varnothing, \varnothing \rangle \).

**Structure operations** permutation \( \times \), duplication \( \triangleleft \) and erasure \( \bullet \) are explicit:

Functions with many outputs:

A list splitting function \( \bigtriangleup \) on lists \( \langle \varnothing, \bigtriangleup \rangle \): both sublists are computed at the same time.
Polygraphic programs

A polygraphic program consists of:
Polygraphic programs

A polygraphic program consists of:

• 1-cells (wires) with 1 composition $\star_0$ to build 1-paths (parallel wires).
Polygraphic programs

A polygraphic program consists of:

- **1-cells** (wires) with 1 composition $\star_0$ to build 1-paths (parallel wires).
- **2-cells** (gates) with 2 compositions $\star_0$ (horizontal) and $\star_1$ (vertical) to build 2-paths (circuits). Three kinds:
  - Structure: $\triangleright$, $\ltimes$, and $\bullet$ with all possible 1-cells.
  - Constructors (with one output) such as $\ominus$, $\ominus$, $\downarrow$, etc.
  - Functions (any possible shape) such as $\triangledown$, $\updownarrow$, $\bullet$, etc.
A **polygraphic program** consists of:

- **1-cells** (wires) with 1 composition $\star_0$ to build 1-paths (parallel wires).

- **2-cells** (gates) with 2 compositions $\star_0$ (horizontal) and $\star_1$ (vertical) to build 2-paths (circuits). Three kinds:
  - Structure: $\rhd$, $\triangledown$ and $\blacklozenge$ with all possible 1-cells.
  - Constructors (with one output) such as $\circ$, $\circlearrowleft$, $\triangledown$, etc.
  - Functions (any possible shape) such as $\rhd$, $\triangledown$, $\blacklozenge$, etc.

- **3-cells** (rules) with 3 compositions $\star_0$, $\star_1$ (parallel) and $\star_2$ (sequential) to build 3-paths (computations). Two kinds:
  - Structure: compute permutations, duplications and erasures on constructors.
  - Computation with source like this ($\varphi$ function and $t$ term):

\[
\begin{align*}
\begin{array}{c}
\vdots \\
t
\end{array} & \star_1 \\
\begin{array}{c}
\vdots \\
\varphi
\end{array} = \\
\begin{array}{c}
\vdots \\
t
\end{array}
\begin{array}{c}
\vdots \\
\varphi
\end{array}
\end{align*}
\]
Computational power of polygraphic programs
Computational power of polygraphic programs

**Theorem:** polygraphic programs form a Turing-complete model of computation.
Computational power of polygraphic programs

**Theorem:** polygraphic programs form a Turing-complete model of computation.

**Proof:** build a polygraphic Turing machine:

\[
q_1 \ a \ \Rightarrow \ q_2 \ b \ \quad \text{if} \ \delta(q_1, a) = (q_2, c, L)
\]

\[
q_1 \ a \ \Rightarrow \ q_2 \ b \ \quad \text{if} \ \delta(q_1, a) = (q_2, c, R)
\]

\[
q_f \ a \ \Rightarrow \ q_0 \ \#
\]
Computational power of polygraphic programs

**Theorem:** polygraphic programs form a Turing-complete model of computation.

**Proof:** build a polygraphic Turing machine:

 Constructors: $\langle \varnothing, \varnothing, \ldots \rangle$ for words.

 Functions: $\bullet$ (computed one) and $\langle q, a \rangle$ (state $q$, reading $a$, inputs are left and right parts of the tape).
Polygraphic programs

Polygraphic interpretations, termination and complexity bounds
**Simplified definition**

**Problem:** find numeric interpretations that prove termination *and* give complexity bounds.
Simplified definition

**Problem:** find numeric interpretations that prove termination *and* give complexity bounds.

**Electrical point of view:** circuits are crossed by currents and gates produce heat.
**Simplified definition**

**Problem:** find numeric interpretations that prove termination *and* give complexity bounds.

**Electrical point of view:** circuits are crossed by currents and gates produce heat.

A *polygraphic interpretation* (p.i.) sends a circuit $f : m \Rightarrow n$ onto monotone maps:

$$f_* = (f_*^1, \ldots, f_*^n) = (\ldots) : \mathbb{N}^m \rightarrow \mathbb{N}^n$$

and

$$[f] = (\ldots) : \mathbb{N}^m \rightarrow \mathbb{N},$$

such that:

- If $x$ is a 1-cell: $x_* = \text{Id}_\mathbb{N}$ and $[x] = 0$.
- If $f$ and $g$ are 2-paths:

$$f \circ_0 g = \begin{array}{c} f \\ \vdots \\ \vdots \\ \vdots \end{array} + \begin{array}{c} g \\ \vdots \\ \vdots \\ \vdots \end{array}$$

$$f \circ_1 g = \begin{array}{c} f \\ \vdots \\ \vdots \\ \vdots \end{array} + \begin{array}{c} g \\ \vdots \\ \vdots \\ \vdots \end{array}$$
Simplified definition

Problem: find numeric interpretations that prove termination and give complexity bounds.

Electrical point of view: circuits are crossed by currents and gates produce heat.

A polygraphic interpretation (p.i.) sends a circuit $f : m \Rightarrow n$ onto monotone maps:

$$f_* = (f_1^*, \ldots, f_n^*) = \left[ \begin{array}{c} f_1 \\ \vdots \\ f_n \end{array} \right] : N^m \to N^n$$

and

$$[f] = \left[ \begin{array}{c} f \end{array} \right] : N^m \to N,$$

such that:

- If $x$ is a 1-cell: $x_* = \text{Id}_N$ and $[x] = 0$.
- If $f$ and $g$ are 2-paths:

$$f_* g = \left[ \begin{array}{c} f \\ g \end{array} \right]$$

$$f_* g = f + g$$

Lemma: the maps $(\cdot)_*$ and $[\cdot]$ are entirely defined by their values on 2-cells.
Simplified definition

**Problem:** find numeric interpretations that prove termination *and* give complexity bounds.

**Electrical point of view:** circuits are crossed by currents and gates produce heat.

A **polygraphic interpretation** (p.i.) sends a circuit $f : m \Rightarrow n$ onto monotone maps:

$$f_* = (f^1_*, \ldots, f^n_*) = f : \mathbb{N}^m \to \mathbb{N}^n$$

and

$$[f] = f : \mathbb{N}^m \to \mathbb{N},$$

such that:

- If $x$ is a 1-cell: $x_* = \text{Id}_\mathbb{N}$ and $[x] = 0$.
- If $f$ and $g$ are 2-paths:

\[
\begin{align*}
\vdots f \vdots g & = \vdots f \vdots + \vdots g \vdots \\
\vdots f \vdots g & = \vdots f \vdots + \vdots g \vdots \\
\vdots f \vdots g & = \vdots f \vdots + \vdots g \vdots \\
\vdots f \vdots g & = \vdots f \vdots + \vdots g \vdots \\
\end{align*}
\]

**Lemma:** the maps $(\cdot)_*$ and $[\cdot]$ are entirely defined by their values on 2-cells.

**Order relation:** $f \triangleright g$ when $f_* \geq g_*$ and $[f] > [g]$.

**Compatibility** with a 3-cell $\alpha : f \Rightarrow g$ when $f \triangleright g$. 

Examples of polygraphic interpretations
Examples of polygraphic interpretations

Standard interpretation of structure 2-cells: $\bowtie_* = (Y, X), \quad \triangle_* = (X, X), \quad \bullet_* = *, \quad$ heats are 0.
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:** \( \begin{bmatrix} \cdot \end{bmatrix} = (Y, X), \quad \begin{bmatrix} \cdot \end{bmatrix} = (X, X), \quad \begin{bmatrix} \cdot \end{bmatrix} = *, \) heats are 0.

**Addition and multiplication:**

Constructors: \( \begin{bmatrix} \cdot \end{bmatrix} = 1, \quad \begin{bmatrix} \cdot \end{bmatrix} = X + 1, \) heats are 0.

Functions: \( \begin{bmatrix} \cdot \end{bmatrix} = X + Y, \quad \begin{bmatrix} \cdot \end{bmatrix} = XY, \quad \begin{bmatrix} \cdot \end{bmatrix} = X, \quad \begin{bmatrix} \cdot \end{bmatrix} = XY + Y. \)
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:** \(\otimes_{*} = (Y, X), \quad \triangledown_{*} = (X, X), \quad \odot_{*} = *, \quad \text{heats are 0.} \)

**Addition and multiplication:**

Constructors: \(\mathfrak{C}_{*} = 1, \quad \mathfrak{C}_{*} = X + 1, \quad \text{heats are 0.} \)

Functions: \(\triangledown_{*} = X + Y, \quad \triangledown_{*} = X Y, \quad \left[\triangledown_{*}\right] = X, \quad \left[\triangledown_{*}\right] = X Y + Y. \)

\[
\begin{cases}
\left[\begin{array}{c}
\triangle \mathfrak{C} \\
\end{array}\right] (X, Y) \\
\left[\begin{array}{c}
\mathfrak{C} \\
\end{array}\right] (X, Y)
\end{cases}
\]

Example:
Examples of polygraphic interpretations

Standard interpretation of structure 2-cells: \( 
\triangleright_\ast = (Y,X), \quad \triangleleft_\ast = (X,X), \quad \bullet_\ast = \ast, \quad \text{heats are 0.}
\)

Addition and multiplication:

Constructors: \( \varnothing_\ast = 1, \quad \vartriangle_\ast = X + 1, \quad \text{heats are 0.} \)

Functions: \( \triangledown_\ast = X + Y, \quad \triangledown_\ast = XY, \quad \begin{bmatrix} \varnothing \end{bmatrix} = X, \quad \begin{bmatrix} \triangledown \end{bmatrix} = XY + Y. \)

\[
\begin{cases}
\begin{bmatrix} \varnothing \end{bmatrix} \quad (X, Y) = 
\begin{bmatrix} \triangledown \end{bmatrix} \quad (X, \vartriangle_\ast (Y))
\end{cases}
\]

Example:

\[
\begin{bmatrix} \triangledown \end{barray} \quad (X, Y)
\]
Examples of polygraphic interpretations

Standard interpretation of structure 2-cells: $\times_* = (Y, X)$, $\triangle_* = (X, X)$, $\circ_* = *$, heats are 0.

Addition and multiplication:

Constructors: $\vartriangleright_* = 1$, $\vartriangleleft_* = X + 1$, heats are 0.

Functions: $\bigtriangledown_* = X + Y$, $\bigtriangleup_* = XY$, $\blacksquare = X$, $\blacktriangle = XY + Y$.

Example:

\[
\begin{align*}
\begin{bmatrix} \vartriangleright \vartriangleleft \end{bmatrix}(X, Y) &= \begin{bmatrix} \bigtriangleup \end{bmatrix}(X, \vartriangleleft_*(Y)) + \begin{bmatrix} \vartriangleleft \end{bmatrix}(Y) \\
\begin{bmatrix} \bigtriangledown \bigtriangleup \end{bmatrix}(X, Y)
\end{align*}
\]
Examples of polygraphic interpretations

Standard interpretation of structure 2-cells: \( \bigtriangleup_*(Y, X), \bigtriangledown_*(X, X), \bigcirc_*=*, \) heats are 0.

Addition and multiplication:

Constructors: \( \bigodot_* = 1, \bigotimes_* = X + 1, \) heats are 0.

Functions: \( \bigtriangledown_* = X + Y, \bigtriangledown_* = XY, [\bigtriangledown] = X, [\bigtriangledown] = XY + Y. \)

\[
\begin{align*}
[\bigcirc] (X, Y) &= [\bigtriangledown] (X, Y + 1) + [\bigotimes] (Y)
\end{align*}
\]

Example:

\[
(\bigtriangledown\bigotimes) (X, Y)
\]
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:** \( \bigotimes_* = (Y, X), \quad \bigtriangleup_* = (X, X), \quad \bigcirc_* = *, \quad \text{heats are 0.} \)

**Addition and multiplication:**

Constructors: \( \bigcirc_* = 1, \quad \bigcirc_* = X + 1, \quad \text{heats are 0.} \)

Functions: \( \bigtriangledown_* = X + Y, \quad \nabla_* = XY, \quad [\bigtriangledown] = X, \quad [\nabla] = XY + Y. \)

\[
\begin{cases}
\begin{array}{c}
\bigtriangledown (X, Y) = X(Y + 1) + (Y + 1) + [\bigcirc] (Y)
\end{array}
\end{cases}
\]

Example:

\[
\begin{cases}
\begin{array}{c}
[\bigtriangleup] (X, Y)
\end{array}
\end{cases}
\]
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:** \( \bigotimes_* = (Y, X) \), \( \bigtriangleup_* = (X, X) \), \( \odot_* = * \), heats are 0.

**Addition and multiplication:**

Constructors: \( \odot_* = 1 \), \( \odot_* = X + 1 \), heats are 0.

Functions: \( \bigtriangledown_* = X + Y \), \( \blacklozenge_* = XY \), \( \bigtriangledown [\bigtriangledown] = X \), \( \bigtriangledown [\blacklozenge] = XY + Y \).

\[
\begin{align*}
\left[\begin{array}{c}
\bigtriangledown \\
\blacklozenge
\end{array}\right] (X, Y) &= X(Y + 1) + (Y + 1) + 0
\end{align*}
\]

Example:

\[
\left[\begin{array}{c}
\bigtriangledown \\
\blacklozenge
\end{array}\right] (X, Y)
\]
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:**\[ \bigtriangleup_{*} = (Y, X), \quad \bigtriangleup_{*} = (X, X), \quad \bigtriangleup_{*} = *, \quad \text{heats are 0.} \]

**Addition and multiplication:**

Constructors: \[ \bigcirc_{*} = 1, \quad \bigcirc_{*} = X + 1, \quad \text{heats are 0.} \]

Functions: \[ \bigstar_{*} = X + Y, \quad \bigstar_{*} = XY, \quad \left[ \bigstar_{*} \right] = X, \quad \left[ \bigstar_{*} \right] = XY + Y. \]

\[
\begin{aligned}
\left[ \begin{array}{c}
\bigcirc \\
\bigcirc
\end{array} \right] (X, Y) &= XY + X + Y + 1, \\
\left[ \begin{array}{c}
\bigtriangleup \\
\bigtriangleup
\end{array} \right] (X, Y) &= \quad (X, Y)
\end{aligned}
\]
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:** \(\bigotimes_* = (Y,X), \bigodot_* = (X,X), \bigcirc_* = \ast\), heats are 0.

**Addition and multiplication:**

Constructors: \(\varnothing_* = 1, \varphi_* = X + 1\), heats are 0.

Functions: \(\bigtriangledown_* = X + Y, \bigtriangledown_* = XY, \begin{bmatrix} \bigtriangledown \end{bmatrix} = X, \begin{bmatrix} \bigtriangledown \end{bmatrix} = XY + Y\).

Example:

\[
\begin{bmatrix} \bigtriangleup \bigcirc \bigtriangledown \end{bmatrix} (X, Y) = \begin{bmatrix} \bigtriangledown \end{bmatrix} \left( \bigodot_* (X), \bigotimes_* \bigotimes_* (X), Y \right)
\]
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:**
\[\mathcal{D}_* = (Y, X), \quad \mathcal{A}_* = (X, X), \quad \mathcal{B}_* = \ast, \quad \text{heats are } 0.\]

**Addition and multiplication:**

Constructors: \(\mathcal{C}_* = 1, \quad \mathcal{D}_* = X + 1, \quad \text{heats are } 0.\)

Functions: \(\mathcal{A}_* = X + Y, \quad \mathcal{B}_* = XY, \quad \mathcal{A} = X, \quad \mathcal{B} = XY + Y.\)

\[
\begin{align*}
\begin{bmatrix} 
\mathcal{A}_* \\
\mathcal{B}_*
\end{bmatrix}(X, Y) &= XY + X + Y + 1,
\end{align*}
\]

Example:

\[
\begin{align*}
\begin{bmatrix} 
\mathcal{C}_* + \\
\mathcal{D}_*
\end{bmatrix}(X, Y) &= \begin{bmatrix} 
\mathcal{A}_*
\end{bmatrix}(\mathcal{A}_*(X), \mathcal{D}_*(\mathcal{A}_*(X), Y)) + \begin{bmatrix} 
\mathcal{B}_*
\end{bmatrix}(\mathcal{A}_*(X), Y)
\end{align*}
\]
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:** \( \triangleright_\ast = (Y, X), \quad \bigtriangleup_\ast = (X, X), \quad \bigcirc_\ast = \ast, \) \( \text{heats are 0}. \)

**Addition and multiplication:**

Constructors: \( \varnothing_\ast = 1, \quad \diamond_\ast = X + 1, \) \( \text{heats are 0}. \)

Functions: \( \nabla_\ast = X + Y, \quad \bigtriangledown_\ast = XY, \quad \bigtriangledown = X, \quad \bigtriangledown = XY + Y. \)

\[
\begin{cases}
\begin{aligned}
\left[ \begin{array}{c}
\bigcirc_\ast \\
\nabla_\ast \\
\end{array} \right] (X, Y) &= XY + X + Y + 1, \\
\left[ \begin{array}{c}
\bigtriangleup_\ast \\
\nabla_\ast \\
\end{array} \right] (X, Y) &= \left[ \nabla_\ast \right] \left( \bigtriangleup_\ast (X), \bigtriangledown_\ast \left( \bigtriangleup_\ast (X), Y \right) \right) + \left[ \nabla_\ast \right] \left( \bigtriangleup_\ast (X), Y \right) + \left[ \bigtriangleup_\ast \right] (X)
\end{aligned}
\end{cases}
\]
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:**\(\circlearrowright_\ast = (Y, X), \quad \circlearrowleft_\ast = (X, X), \quad \bullet_\ast = \ast, \quad \text{heats are } 0.\)

**Addition and multiplication:**

Constructors: \(\circ_\ast = 1, \quad \bullet_\ast = X + 1, \quad \text{heats are } 0.\)

Functions: \(\bigtriangledown_\ast = X + Y, \quad \bigtriangleup_\ast = XY, \quad \left[\bigtriangledown\right] = X, \quad \left[\bigtriangleup\right] = XY + Y.\)

\[
\begin{cases}
\left[\bigtriangledown\bigcirc\right] (X, Y) = XY + X + Y + 1, \\
\left[\bigtriangleup\bigcirc\right] (X, Y) = \left[\bigtriangledown\right] (X, \bigtriangleup\ast (X, Y)) + \left[\bigtriangleup\right] (X, Y) + 0
\end{cases}
\]
Examples of polygraphic interpretations

Standard interpretation of structure 2-cells: \( \bigstar_\ast = (Y,X), \quad \triangle_\ast = (X,X), \quad \bullet_\ast = \ast, \) heats are 0.

Addition and multiplication:

Constructors: \( \bigcirc_\ast = 1, \quad \bigcirc_\ast = X + 1, \) heats are 0.

Functions: \( \bigtriangledown_\ast = X + Y, \quad \bigtriangledown_\ast = XY, \quad \begin{bmatrix} \bigtriangleup \end{bmatrix} = X, \quad \begin{bmatrix} \bigtriangledown \end{bmatrix} = XY + Y. \)

\[
\begin{cases}
\begin{bmatrix} \bigtriangledown \end{bmatrix} (X, Y) = XY + X + Y + 1,
\end{cases}
\]

Example:

\[
\begin{cases}
\begin{bmatrix} \bigtriangleup \bigtriangledown \end{bmatrix} (X, Y) = X + \begin{bmatrix} \bigtriangledown \end{bmatrix} (X, Y)
\end{cases}
\]
Examples of polygraphic interpretations

Standard interpretation of structure 2-cells: \( \bigcirc = (Y,X), \quad \bigtriangleup = (X,X), \quad \bullet = *, \quad \text{heats are } 0. \)

Addition and multiplication:

Constructors: \( \mathcal{Q} = 1, \quad \mathcal{P} = X + 1, \quad \text{heats are } 0. \)

Functions: \( \mathcal{P} = X + Y, \quad \mathcal{Q} = XY, \quad \mathcal{M} = X, \quad \mathcal{Q} = XY + Y. \)

Example:

\[
\begin{align*}
\begin{bmatrix}
\mathcal{Q} & \mathcal{P} \\
\end{bmatrix}(X, Y) &= XY + X + Y + 1, \\
\begin{bmatrix}
\mathcal{Q} & \mathcal{P} \\
\end{bmatrix}(X, Y) &= X + (X + 1)Y
\end{align*}
\]
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:** $\star = (Y, X), \quad \triangledown = (X, X), \quad \varnothing = *, \quad$ heats are $0$.

**Addition and multiplication:**

 Constructors: $\circ = 1, \quad \varnothing = X + 1, \quad$ heats are $0$.

 Functions: $\nabla = X + Y, \quad \nabla = XY, \quad [\nabla] = X, \quad [\nabla] = XY + Y$.

\[
\begin{cases}
\begin{bmatrix}
\circ
\end{bmatrix}(X, Y) = XY + X + Y + 1, \\
\begin{bmatrix}
\varnothing
\end{bmatrix}(X, Y) = XY + X + Y.
\end{cases}
\]
Examples of polygraphic interpretations

Standard interpretation of structure 2-cells: \( 
\begin{array}{c c}
\downarrow & = (Y, X) \\
\bigtriangleup & = (X, X) \\
\circ & = * \\
\end{array} 
\) 
heats are 0.

Addition and multiplication:

 Constructors: \( \circ = 1 \), \( \circ = X + 1 \), heats are 0.

Functions: \( \bigtriangleup = X + Y \), \( \bigtriangledown = XY \), \( \left\lceil X \right\rceil = X \), \( \left\lfloor X \right\rfloor = X + Y + 1 \).

Example:

\[
\begin{cases}
\left\lceil X \right\rceil (X, Y) = XY + X + Y + 1, \\
\end{cases}
\]

List splitting: \( \circ = 1 \), \( \bigtriangledown = X + Y + 1 \), \( \bigtriangleup = \left( \left\lceil \frac{X}{2} \right\rceil, \left\lfloor \frac{X}{2} \right\rfloor \right) \), \( \left\lfloor X \right\rfloor = X \).
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:** \( \otimes_* = (Y,X), \ \bigtriangleup_* = (X,X), \ \odot_* = *, \ \text{heats are } 0. \)

**Addition and multiplication:**

 Constructors: \( \varnothing_* = 1, \ \varnothing_* = X+1, \ \text{heats are } 0. \)

 Functions: \( \downarrow_* = X+Y, \ \uparrow_* = XY, \ \lfloor \downarrow \rfloor = X, \ \lceil \downarrow \rceil = XY+Y. \)

\[
\begin{aligned}
\left[ \begin{array}{c}
\downarrow \\
\end{array} \right] (X,Y) &= XY + X + Y + 1, \\
\left[ \begin{array}{c}
\downarrow \\
\end{array} \right] (X,Y) &= XY + X + Y.
\end{aligned}
\]

**Example:**

\[
\left[ \begin{array}{c}
\bigtriangleup \\
\end{array} \right] (X,Y) = XY + X + Y.
\]

**List splitting:** \( \varnothing_* = 1, \ \downarrow_* = X + Y + 1, \ \bigtriangleup_* = \left( \left\lfloor \frac{X}{2} \right\rfloor, \left\lceil \frac{X}{2} \right\rceil \right), \ \lceil \bigtriangleup \rceil = X. \)

**Integer division:** minus is \( \downarrow \) and division is \( \downarrow \)

 Constructors: \( \varnothing_* = 1, \ \varnothing_* = X+1, \ \text{heats are } 0. \)

 Functions: \( \downarrow_* = X, \ \uparrow_* = X, \ \lfloor \downarrow \rfloor = Y + 1, \ \lceil \downarrow \rceil = XY + 2X. \)
Examples of polygraphic interpretations

**Standard interpretation of structure 2-cells:** \( \triangleright_2 = (Y, X), \quad \triangleleft_2 = (X, X), \quad \triangleright_2 = *, \) heats are 0.

**Addition and multiplication:**

Constructors: \( \circ_* = 1, \quad \circ_* = X + 1, \) heats are 0.

Functions: \( \sqsubseteq_2 = X + Y, \quad \sqcup_2 = XY, \quad \lfloor_2 \rfloor = X, \quad \lceil_2 \rceil = XY + Y. \)

Example:

\[
\left\lfloor \begin{array}{c}
\circ_2 \\
\sqcup_2 \\
\lceil_2 \\
\end{array} \right\rfloor_2 (X, Y) = XY + X + Y + 1,
\]

**List splitting:** \( \circ_* = 1, \quad \sqsubseteq_2 = X + Y + 1, \quad \lceil_2 \rceil = (\lceil \frac{X}{2} \rceil, \lceil \frac{X}{2} \rceil), \quad \lfloor_2 \rfloor = X. \)

**Integer division:** minus is \( \sqsubseteq_2 \) and division is \( \sqcup_2 \)

Constructors: \( \circ_* = 1, \quad \circ_* = X + 1, \) heats are 0.

Functions: \( \sqsubseteq_2 = X, \quad \sqcup_2 = X, \quad \lfloor_2 \rfloor = Y + 1, \quad \lceil_2 \rceil = XY + 2X. \)

Compatible with all computation rules: order \( \succ \) is not a simplification order.
Termination results

**Theorem:** if a polygraphic program admits a p.i. compatible with its 3-cells, then it terminates.
Termination results

Theorem: if a polygraphic program admits a p.i. compatible with its 3-cells, then it terminates.

A polygraphic program is simple when there is a p.i. "such as in the examples" and compatible with its computation 3-cells.
Termination results

**Theorem:** if a polygraphic program admits a p.i. compatible with its 3-cells, then it terminates.

A polygraphic program is **simple** when there is a p.i. "such as in the examples" and compatible with its *computation* 3-cells.

**Theorem:** simple polygraphic programs terminate.
**Termination results**

**Theorem:** if a polygraphic program admits a p.i. compatible with its 3-cells, then it terminates.

A polygraphic program is **simple** when there is a p.i. "such as in the examples" and compatible with its *computation* 3-cells.

**Theorem:** simple polygraphic programs terminate.

**Proof:**

– p.i. satisfies $f \succeq g$ for each structure 3-cell $\alpha : f \Rightarrow g$;

– hence: program terminates iff structure 3-cells terminate;

– structure 3-cells terminate (with *structure heat map*).
Spatial complexity of simple programs
Spatial complexity of simple programs

**Question:** can we bound spatial complexity, *i.e.* the size $||u||$ of any intermediate value $u$ produced by a computation?
**Spatial complexity of simple programs**

**Question:** can we bound spatial complexity, *i.e.* the size $\|u\|$ of any intermediate value $u$ produced by a computation?

**Answer:** yes, with currents maps.
Spatial complexity of simple programs

**Question:** can we bound spatial complexity, *i.e.* the size $||u||$ of any intermediate value $u$ produced by a computation?

**Answer:** yes, with currents maps.

**Lemma:** for every value $u$, we have $||u|| \leq u_\times \leq \alpha ||u||$, where $\alpha$ is a global positive constant, given by the current maps of the constructors.
Spatial complexity of simple programs

**Question:** can we bound spatial complexity, *i.e.* the size $||u||$ of any intermediate value $u$ produced by a computation?

**Answer:** yes, with currents maps.

**Lemma:** for every value $u$, we have $||u|| \leq u_* \leq a||u||$, where $a$ is a global positive constant, given by the current maps of the constructors.

**Proposition:** for every function $\varphi$ with $n$ inputs and every family $u = (u_1, \ldots, u_n)$ of values:

$$ ||\varphi(u)|| \leq P_{\varphi}(||u_1||, \ldots, ||u_n||), \quad \text{where} \quad P_{\varphi} = \sum_{j=1}^{n} \varphi_j^* (aX_1, \ldots, aX_n). $$

The inequality also holds if $\varphi(u)$ is replaced by any intermediate value.
Spatial complexity of simple programs

**Question:** can we bound spatial complexity, *i.e.* the size $||u||$ of any intermediate value $u$ produced by a computation?

**Answer:** yes, with currents maps.

**Lemma:** for every value $u$, we have $||u|| \leq u_x \leq a ||u||$, where $a$ is a global positive constant, given by the current maps of the constructors.

**Proposition:** for every function $\varphi$ with $n$ inputs and every family $u = (u_1, \ldots, u_n)$ of values:

$$||\varphi(u)|| \leq P_\varphi(||u_1||, \ldots, ||u_n||), \quad \text{where} \quad P_\varphi = \sum_{j=1}^{n} \varphi_j^1(aX_1, \ldots, aX_n).$$

The inequality also holds if $\varphi(u)$ is replaced by any intermediate value.

**Examples:**

- **Addition and multiplication:** $P_{\oplus} = X + Y$ and $P_{\times} = XY$ ($\sim 2X + Y$ and $\sim XY + X + Y$ with terms).
- **Division:** $P_{\bigtriangledown} = X$ and $P_{\div} = X$ (no bound with terms).
- **List splitting:** $P_{\triangleright} = \bigtriangleup_1^1 + \bigtriangleup_2^2 = \lceil \frac{X}{2} \rceil + \lfloor \frac{X}{2} \rfloor = X$ ($\sim 2X$ with terms).
Temporal complexity of simple programs
Temporal complexity of simple programs

**Idea:** the heat maps bound temporal complexity,
i.e. the number $|||F|||$ of computation 3-cells in any normalizing 3-path $F$. 
Temporal complexity of simple programs

Idea: the heat maps bound temporal complexity, 
*i.e.* the number $|||F|||$ of *computation* 3-cells in any normalizing 3-path $F$.

Proposition: for every function $\varphi$, every family $u = (u_1, \ldots, u_n)$ of values and every normalizing path $F : u \star \varphi \Rightarrow \varphi(u)$, we have $|||F||| \leq Q_\varphi(|||u_1|||, \ldots, |||u_n|||)$, where $Q_\varphi = [\varphi](aX_1, \ldots, aX_n)$. 
Temporal complexity of simple programs

Idea: the heat maps bound temporal complexity, i.e. the number $||| F |||$ of computation 3-cells in any normalizing 3-path $F$.

Proposition: for every function $\varphi$, every family $u = (u_1, \ldots, u_n)$ of values and every normalizing path $F : u \star 1 \varphi \Rightarrow \varphi(u)$, we have $||| F ||| \leq Q_{\varphi}(|||u_1|||, \ldots, |||u_n|||)$, where $Q_{\varphi} = [\varphi](aX_1, \ldots, aX_n)$.

Examples:

- Addition and multiplication: $Q_\sum = X$ and $Q_\prod = XY + Y$ (≈ $2X + Y$ and $\sim XY + X + Y$ with terms).
- Division: $Q_\div = Y + 1$ and $Q_\div = XY + 2X$ (no bound with terms).
- List splitting: $Q_\tri = X$ (≈ $2X$ with terms).
Temporal complexity of simple programs

Idea: the heat maps bound temporal complexity, 
*i.e.* the number $|||F|||$ of computation 3-cells in any normalizing 3-path $F$.

**Proposition:** for every function $\varphi$, every family $u = (u_1, \ldots, u_n)$ of values and every normalizing path $F : u \star_1 \varphi \Rightarrow \varphi(u)$, we have $|||F||| \leq Q_\varphi(||u_1||, \ldots, ||u_n||)$, where $Q_\varphi = \{\varphi\}(aX_1, \ldots, aX_n)$.

**Examples:**

- Addition and multiplication: $Q_\varphi = X$ and $Q_\varphi = XY + Y$  ($\sim 2X + Y$ and $\sim XY + X + Y$ with terms).
- Division: $Q_\varphi = Y + 1$ and $Q_\varphi = XY + 2X$  (no bound with terms).
- List splitting: $Q_\varphi = X$  ($\sim 2X$ with terms).

**Theorem:** simple programs compute exactly PTIME functions.
Temporal complexity of simple programs

Idea: the heat maps bound temporal complexity, i.e. the number $\|\|F\|\|$ of computation 3-cells in any normalizing 3-path $F$.

Proposition: for every function $\varphi$, every family $u = (u_1, \ldots, u_n)$ of values and every normalizing path $F: u \star_1 \varphi \Rightarrow \varphi(u)$, we have $\|\|F\|\| \leq Q_{\varphi}(\|u_1\|, \ldots, \|u_n\|)$, where $Q_{\varphi} = [\varphi](aX_1, \ldots, aX_n)$.

Examples:

- Addition and multiplication: $Q_{\varnothing} = X$ and $Q_{\varphi} = XY + Y \ (\sim 2X + Y \text{ and } \sim XY + X + Y \text{ with terms})$.
- Division: $Q_{\varnothing} = Y + 1$ and $Q_{\varphi} = XY + 2X \ (\text{no bound with terms})$.
- List splitting: $Q_{\varphi} = X \ (\sim 2X \text{ with terms})$.

Theorem: simple programs compute exactly PTIME functions.

Proof:

- use above proposition to bound computation 3-cells by a polynomial;
- build a polynomial bound for number of structure 3-cells;
- prove that each rewriting step takes polynomial time;
- build a polygraphic Turing machine with a clock.