University of Amsterdam

Bachelor Informatica

Thread-Oriented Program algebra

Author: Sebastian Melzer
Supervisors: Alban Ponse and Inge Bethke

June, 2018

Context

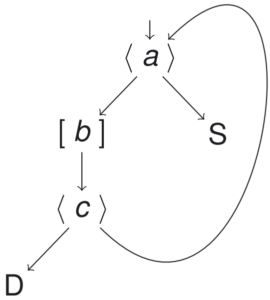
- Basic Thread Algebra (BTA)
  - Syntax to define the **behaviour** of sequential programs
- Program Algebra (PGA)
  - Simple program notation (language)
  - Uses BTA to define its behaviour
- Semigroup $\mathcal{C}$
  - Alternative to PGA
  - No **directional bias**
- Thread-Oriented Program algebra (TOP)
  - Variation of $\mathcal{C}$
  - Strong correlation with BTA

Basic Thread Algebra

- Used to describe the behaviour of sequential programs
- Arbitrary set of actions $\mathcal{A} = \{a, b, c, ...\}$
- All actions yield `true` or `false` on execution
- BTA expressions are called threads $\{P, Q, R, \ldots\}$
  - The *deadlock* constant D
  - The *termination* constant S
  - The *postconditional composition* operator $P \trianglelefteq a \trianglerighteq Q$
    - The *action prefix* operator $a \circ P$ is shorthand for $P \trianglelefteq a \trianglerighteq P$

Example

$$P_1 = P_2 \unlhd a \unrhd P_4$$
$$P_2 = b \circ P_3$$
$$P_3 = P_5 \unlhd c \unrhd P_1$$
$$P_4 = \mathrm{S}$$
$$P_5 = \mathrm{D}$$



TOP syntax

Instruction sequence

A finite sequence of instructions: $u_1; u_2; \ldots; u_n$

Instructions

- *Basic* instructions $/a$ and $\backslash a$ for $a \in \mathcal{A}$
- *Jump* instructions $/\#k$ and $\backslash\#k$ for $k \in \mathbb{N}$
- *Test* instructions $+a$ and $-a$ for $a \in \mathcal{A}$
- The *termination* instruction $!$
- The *abort* instruction $\#$

TOP semantics

For an instruction sequence $X = u_1 ; \ldots ; u_n$, the thread extraction operator $|X|_i$ returns the regular thread that is modelled by u_i in X.

Definition

$$|X|_i = \begin{cases} a \circ |X|_{i+1} & \text{if } u_i = /a, \\ |X|_{i+k} & \text{if } u_i = /\#k, \\ |X|_{i-1} \trianglelefteq a \trianglerighteq |X|_{i+1} & \text{if } u_i = +a, \\ \vdots \quad \text{similar equations for } \backslash a, \backslash\#k, -a & \\ \mathsf{D} & \text{if } u_i = \#, \\ \mathsf{S} & \text{if } u_i = !. \end{cases}$$

Example

$$X = /\#2; !; +a; -b; \#$$

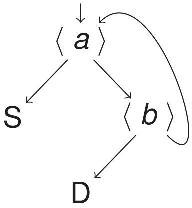
$$|X|_1 = |X|_3$$

$$|X|_2 = \mathsf{S}$$

$$|X|_3 = |X|_2 \trianglelefteq a \trianglerighteq |X|_4$$

$$|X|_4 = |X|_5 \trianglelefteq b \trianglerighteq |X|_3$$

$$|X|_5 = \mathsf{D}$$



On the length of TOP instruction sequences

(Q1) Instruction sequence to thread

How many linear equations are required to define the behaviour expressed by an instruction sequence of length n?

(Q2) Thread to instruction sequence

How many instructions are required to express the behaviour defined by a linear specification of n equations?

Q1: Instruction sequence to thread

Theorem

The behaviour expressed by a TOP *instruction sequence of length n can be defined by a linear specification of $n + 1$ equations.*

Proof.

1. There are three base cases:
   - Basic or test instruction results in 2 equations
   - Jump, abort, or termination instruction results in 1 equation
2. To add an instruction to the sequence X we need at most 1 extra equation.

□

Example

Instruction sequence	Linear specification
$+a$	$P_1 = a \circ P_D$ $P_D = D$
$+a; /b$	$P_1 = P_D \trianglelefteq a \trianglerighteq P_2$ $P_2 = b \circ P_D$ $P_D = D$
$+a; /b; !$	$P_1 = P_D \trianglelefteq a \trianglerighteq P_2$ $P_2 = b \circ P_3$ $P_3 = S$ $P_D = D$

Q2: Thread to instruction sequence

Theorem

The minimum length of TOP *instruction sequences required to model all regular threads of n states is $3n - \lceil \frac{n}{2} \rceil$.*

Proof.

1. Showing that $3n$ is an upper bound is trivial
2. For a constant (S or D) 2 instructions can be saved.
3. When a state is directly reachable from two other states (or twice from the same state) an instruction can be saved.
   - There are at least $\lceil \frac{n}{2} \rceil$ such cases if no state is constant.

□

On the expressiveness of TOP

(Q3) Size of jumps counters

Are arbitrarily large jump counters required in both directions to model all regular threads?

Q3: Size of jump counters

Theorem

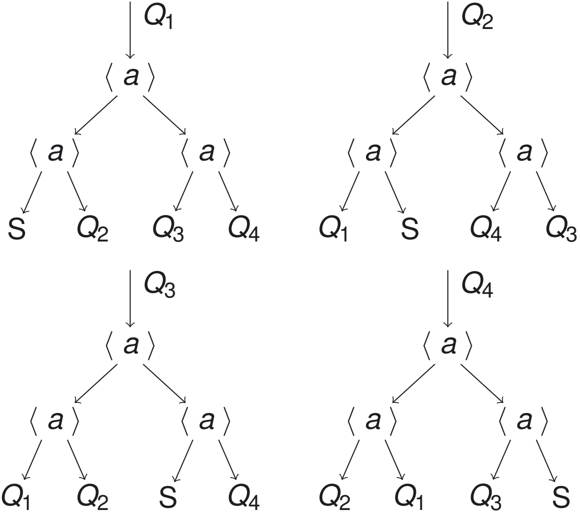
Let $\text{TOP}^{\leq k}$ be the subset of TOP that includes all instructions except forward jump instructions with a jump counter greater than $k \in \mathbb{N}^{+}$.

$\text{TOP}^{\leq k}$ instruction sequences cannot model all regular threads for any k.

Proof.

Assuming the opposite results in a contradiction. □

Example



TOP in 2 dimensions

Instruction plane

A matrix of instructions: $X = \begin{bmatrix} u_{1,1} & \cdots & u_{1,n} \\ \vdots & \ddots & \vdots \\ u_{m,1} & \cdots & u_{m,n} \end{bmatrix}$.

Instructions

- *Basic* instructions $/a$, $\backslash a$, $\uparrow a$, $\downarrow a$ for $a \in \mathcal{A}$
- *Jump* instructions $/\#k$, $\backslash\#k$, $\uparrow\#k$, $\downarrow\#k$ for $k \in \mathbb{N}$
- *Test* instructions $+a$, $-a$, $\updownarrow{+a}$, $\updownarrow{-a}$ for $a \in \mathcal{A}$
- The *termination* instruction $!$
- The *abort* instruction $\#$

Properties of TOP_2

(Q4) Expressiveness TOP versus TOP_2

Does the second dimension of TOP_2 add expressiveness in comparison to TOP?

(Q5) Size of jumps counters in 2D

Are arbitrarily large jump counters required in TOP_2 to model all regular threads?

Q4: Expressiveness TOP versus TOP_2

Theorem

TOP *instruction sequences and* TOP_2 *instruction planes are equally expressive.*

Proof.

1. An instruction sequence can be transformed into an instruction plane.
2. An instruction plane can be transformed into an instruction sequence.

□

Q5: Size of jumps counters in 2D

Theorem

Let $\text{TOP}_2^{\leq 2}$ be the subset of TOP_2 that does not contain jump instructions with a jump counter greater than 2.

$\text{TOP}_2^{\leq 2}$ instruction sequences can model all regular threads.

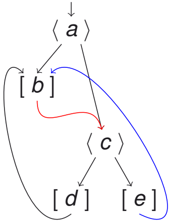
Proof.

1. If P is a regular thread you can draw a 2-dimensional representation of P.
2. The image can be discretised into an instruction plane which models P.

□

Example

$$P_1 = P_2 \trianglelefteq a \trianglerighteq P_3$$
$$P_2 = b \circ P_3$$
$$P_3 = P_4 \trianglelefteq c \trianglerighteq P_5$$
$$P_4 = d \circ P_2$$
$$P_5 = e \circ P_2$$



$$\begin{bmatrix}
 & & \downarrow\#1 & +a & \downarrow\#1 & & & & \\
 & \downarrow\#1 & \backslash\#1 & & \downarrow\#1 & & & & \\
/\#1 & \downarrow\#1 & \backslash\#1 & \backslash\#1 & \downarrow\#1 & \backslash\#2 & \backslash\#1 & & \\
\uparrow\#1 & \downarrow b & & & /\#1 & \downarrow\#1 & \uparrow\#1 & \backslash\#1 & \\
\uparrow\#1 & \downarrow\#1 & & & & \downarrow\#1 & & \uparrow\#1 & \\
\uparrow\#1 & /\#1 & /\#1 & /\#1 & /\#1 & \downarrow\#1 & & \uparrow\#1 & \backslash\#1 \\
\uparrow\#1 & & & & \downarrow\#1 & +c & \downarrow\#1 & & \uparrow\#1 \\
\uparrow\#1 & & & \downarrow\#1 & \backslash\#1 & & /\#1 & \downarrow\#1 & \uparrow\#1 \\
\uparrow\#1 & \backslash\#1 & & \downarrow\#1 & & & & \downarrow\#1 & \uparrow\#1 \\
 & \uparrow\#1 & & \downarrow d & & & & \downarrow e & \uparrow\#1 \\
 & \uparrow\#1 & \backslash\#1 & \backslash\#1 & & & & /\#1 & \uparrow\#1
\end{bmatrix}$$

Discussion

Summary

- TOP to BTA: n instructions $\rightarrow$ $n+1$ states
- BTA to TOP: n states $\rightarrow$ $3n - \lceil \frac{n}{2} \rceil$ instructions
- TOP and TOP_2 are equally expressive
- Arbitrarily large jumps necessary in TOP but not in TOP_2

Questions?

$$\begin{bmatrix}
 & /\#1 & /e & /\#1 & \downarrow s & \\
\downarrow q & \uparrow\#1 & & & /\#1 & \downarrow\#1 \\
/\#1 & \uparrow u & & & \downarrow\#2 & \downarrow t \\
 & & & & \downarrow i & \backslash\#1 \\
 & & & \downarrow o & \backslash\#1 & \\
 & & \downarrow\#3 & \downarrow\#1 & & \\
 & & \uparrow\#1 & \backslash n & & \\
 & & & & & \\
 & & /s & \downarrow\#1 & & \\
 & & ! & \backslash\#1 & &
\end{bmatrix}$$

Why is TOP thread-oriented?

Language	Instruction sequence	Thread
PGA	$+a; u_1; u_2$	$\|u_1\| \unlhd a \unrhd \|u_2\|$
$\mathcal{C}$	$/+a; u_1; u_2$	$\|u_1\| \unlhd a \unrhd \|u_2\|$
TOP	$u_1; +a; u_2$	$\|u_1\| \unlhd a \unrhd \|u_2\|$

Why is $3n$ an upper bound?

Transforming $\{P_1{=}t_1, \ldots, P_n{=}t_n\}$ into $X = X_1; \cdots; X_n$ results in at most three instructions per X_i.

$$X_i = \begin{cases} ! & \text{if } P_i = \mathrm{S}, \\ \# & \text{if } P_i = \mathrm{D}, \\ \mathcal{J}(P_j); +a; \mathcal{J}(P_k) & \text{if } P_i = P_j \trianglelefteq a \trianglerighteq P_k, \end{cases}$$

where $\mathcal{J}(P_j)$ is a jump instruction to X_j.

Example

$$\begin{aligned} P_1 &= P_2 \trianglelefteq a \trianglerighteq P_3 \\ P_2 &= \mathrm{D} \\ P_3 &= b \circ P_4 \\ P_4 &= \mathrm{S} \end{aligned} \quad \rightarrow \quad \begin{aligned} X_1 &= \mathcal{J}(P_2); +a; \mathcal{J}(P_3) \\ X_2 &= \# \\ X_3 &= \mathcal{J}(P_4); +b; \mathcal{J}(P_4) \\ X_4 &= ! \end{aligned}$$

$$X = /\#3; +a; /\#3; \#; /\#3; +b; /\#1; !$$

Why are there at least $\lceil \frac{n}{2} \rceil$ cases?

If we consider only non-constant states:

1. Each state has two *outbound arcs*. ($P \unlhd a \unrhd Q$)
2. If there are n states, there are $2n$ arcs in total.
3. Each state that has a pair of *inbound arcs* is such a case.
4. Minimising this number of pairs results in $\lceil \frac{n}{2} \rceil$.

Example

If $n = 6$ there at least 3 such pairs.

State	P_1	P_2	P_3	P_4	P_5	P_6
Outbound arcs	2	2	2	2	2	2
Inbound arcs	1	$\underline{3}$	1	$\underline{3}$	1	$\underline{3}$