Simple Stream Specifications with Undecidable Productivity

Bartek Klin and Beata Nachyła

Eindhoven, April 2, 2016

Stream specifications

$$\mathtt{z} \longrightarrow \mathtt{0} : \mathtt{z} \quad \mathtt{o} \longrightarrow \mathtt{1} : \mathtt{o}$$

$$\mathtt{zip}(x : \sigma, \tau) \longrightarrow x : \mathtt{zip}(\tau, \sigma)$$

Stream specifications

$$\mathtt{z} \longrightarrow \mathtt{0} : \mathtt{z} \quad \mathtt{o} \longrightarrow \mathtt{1} : \mathtt{o}$$
$$\mathtt{zip}(x : \sigma, \tau) \longrightarrow x : \mathtt{zip}(\tau, \sigma)$$

$$\mathtt{zip}(\mathtt{z}, \mathtt{o}) \to \mathtt{zip}(\mathtt{0} : \mathtt{z}, \mathtt{o}) \to \mathtt{0} : \mathtt{zip}(\mathtt{o}, \mathtt{z})$$
$$\to \mathtt{0} : \mathtt{zip}(\mathtt{1} : \mathtt{o}, \mathtt{z}) \to \mathtt{0} : \mathtt{1} : \mathtt{zip}(\mathtt{z}, \mathtt{o}) \to \cdots$$

Stream specifications

$$\mathtt{z} \longrightarrow 0 : \mathtt{z} \quad \mathtt{o} \longrightarrow 1 : \mathtt{o}$$
$$\mathtt{zip}(x : \sigma, \tau) \longrightarrow x : \mathtt{zip}(\tau, \sigma)$$

$$\mathtt{zip}(\mathtt{z}, \mathtt{o}) \to \mathtt{zip}(0 : \mathtt{z}, \mathtt{o}) \to 0 : \mathtt{zip}(\mathtt{o}, \mathtt{z})$$
$$\to 0 : \mathtt{zip}(1 : \mathtt{o}, \mathtt{z}) \to 0 : 1 : \mathtt{zip}(\mathtt{z}, \mathtt{o}) \to \cdots$$

Stream specifications

$$\mathtt{z} \longrightarrow 0 : \mathtt{z} \quad \mathtt{o} \longrightarrow 1 : \mathtt{o}$$
$$\mathtt{zip}(x : \sigma, \tau) \longrightarrow x : \mathtt{zip}(\tau, \sigma)$$

$$\mathtt{zip}(\mathtt{z}, \mathtt{o}) \to \mathtt{zip}(0 : \mathtt{z}, \mathtt{o}) \to 0 : \mathtt{zip}(\mathtt{o}, \mathtt{z})$$
$$\to 0 : \mathtt{zip}(1 : \mathtt{o}, \mathtt{z}) \to 0 : 1 : \mathtt{zip}(\mathtt{z}, \mathtt{o}) \to \cdots$$

$$\mathtt{c} \longrightarrow 0 : \mathtt{f}(\mathtt{c})$$
$$\mathtt{f}(x : y : \sigma) \longrightarrow x : \mathtt{f}(\sigma)$$

Stream specifications

$$\mathtt{z} \longrightarrow 0 : \mathtt{z} \quad \mathtt{o} \longrightarrow 1 : \mathtt{o}$$
$$\mathtt{zip}(x : \sigma, \tau) \longrightarrow x : \mathtt{zip}(\tau, \sigma)$$

$$\mathtt{zip}(\mathtt{z}, \mathtt{o}) \to \mathtt{zip}(0 : \mathtt{z}, \mathtt{o}) \to 0 : \mathtt{zip}(\mathtt{o}, \mathtt{z})$$
$$\to 0 : \mathtt{zip}(1 : \mathtt{o}, \mathtt{z}) \to 0 : 1 : \mathtt{zip}(\mathtt{z}, \mathtt{o}) \to \cdots$$

$$\mathtt{c} \longrightarrow 0 : \mathtt{f}(\mathtt{c})$$
$$\mathtt{f}(x : y : \sigma) \longrightarrow x : \mathtt{f}(\sigma)$$

$$\mathtt{c} \to 0 : \mathtt{f}(\mathtt{c}) \to 0 : \mathtt{f}(0 : \mathtt{f}(\mathtt{c})) \to 0 : \mathtt{f}(0 : \mathtt{f}(0 : \mathtt{f}(\mathtt{c})))$$

Limits of decidability

Stream differential equations(C.Kupke, M.Niqui, J.Rutten, 2011)

$$i(\sigma, \tau) := \sigma \qquad d(a1.a2) := \mathtt{zip}(y2, x1)$$

Limits of decidability

Stream differential equations(C.Kupke, M.Niqui, J.Rutten, 2011)

$$i(\sigma, \tau) := \sigma \qquad d(a1.a2) := \mathtt{zip}(y2, x1)$$

Pure stream specifications(J.Endrullis, D.Hendriks, J.W.Klop, 2013)

$$\mathtt{S} \to \mathtt{q}(0 : 1 : \mathtt{S}) \qquad \mathtt{q}(0 : \sigma) \to 0 : 1 : \mathtt{q}(\sigma) \qquad \mathtt{q}(1 : \sigma) \to 1 : 0 : q(\sigma)$$

Limits of decidability

Stream differential equations(C.Kupke, M.Niqui, J.Rutten, 2011)

$$i(\sigma, \tau) := \sigma \qquad d(a1.a2) := \mathtt{zip}(y2, x1)$$

Pure stream specifications(J.Endrullis, D.Hendriks, J.W.Klop, 2013)

$$\mathtt{S} \to \mathtt{q}(0 : 1 : \mathtt{S}) \qquad \mathtt{q}(0 : \sigma) \to 0 : 1 : \mathtt{q}(\sigma) \qquad \mathtt{q}(1 : \sigma) \to 1 : 0 : q(\sigma)$$

Lazy Stream Specification (J.Endrullis, C.Grabmayer, D.Hendriks, 2009)

$$M_P \longrightarrow \mathtt{zip_d}(T_1, \ldots, T_d), \text{ where:}$$

$$T_n = \begin{cases} \mathtt{mod_{p'_n}}(\mathtt{tail}^{b_{n1}}(\mathtt{M_P})), \text{ if } p'_n \text{ is defined,} \\ \bullet : \quad \mathtt{mod}_{\mathtt{d}}(\mathtt{tail}^{n-1}(\mathtt{M_P})), \text{ if } p'_n \text{ is undefined} \end{cases}$$

$$\mathtt{head}(x : \sigma) \longrightarrow x \quad \mathtt{tail}(x : \sigma) \longrightarrow \sigma$$

$$\mathtt{mod_k}(\sigma) \longrightarrow \mathtt{head}(\sigma) : \mathtt{mod_k}(\mathtt{tail_k}(\sigma))$$

$$\mathtt{zip_d}(\sigma_1, \sigma_2, \ldots, \sigma_d) \longrightarrow \mathtt{head}(\sigma_1) : \mathtt{zip_d}(\sigma_2, \ldots, \sigma_d, \mathtt{tail}(\sigma_1))$$

Queue machines

$QM = \langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$

- Like pushdown automata, but:

Queue machines

$QM = \langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$

- Like pushdown automata, but:
  - with a queue instead of a stack

Queue machines

$QM = \langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$

- Like pushdown automata, but:
  - with a queue instead of a stack
  - a_1 is and input word and is initially stored in the queue

Queue machines

$QM = \langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$

- Like pushdown automata, but:
  - with a queue instead of a stack
  - a_1 is and input word and is initially stored in the queue
- Our variant:

$$\delta_0 : Q \rightharpoonup A \times Q$$
$$\delta_1 : Q \times A \rightharpoonup A \times Q$$
$$\delta_2 : Q \times A^2 \rightharpoonup A \times Q$$

Queue machines

$QM = \langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$

- Like pushdown automata, but:
  - with a queue instead of a stack
  - a_1 is and input word and is initially stored in the queue
- Our variant:

$$\delta_0 : Q \rightharpoonup A \times Q$$
$$\delta_1 : Q \times A \rightharpoonup A \times Q$$
$$\delta_2 : Q \times A^2 \rightharpoonup A \times Q$$

- Halts by taking 2 letters where there is 1.

Queue machines

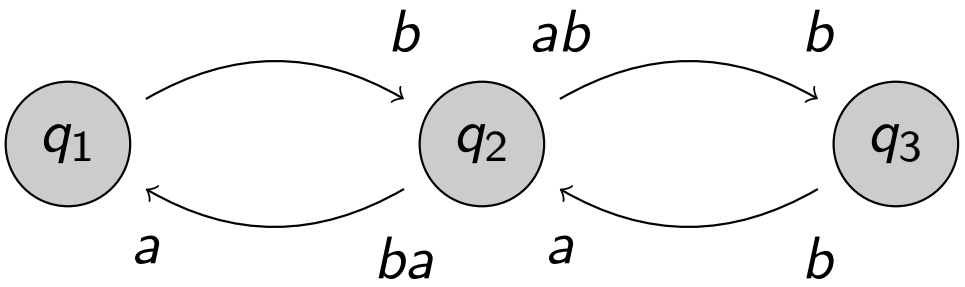
$QM = \langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$

- Like pushdown automata, but:
  - with a queue instead of a stack
  - a_1 is and input word and is initially stored in the queue
- Our variant:

$$\delta_0 : Q \rightharpoonup A \times Q$$
$$\delta_1 : Q \times A \rightharpoonup A \times Q$$
$$\delta_2 : Q \times A^2 \rightharpoonup A \times Q$$

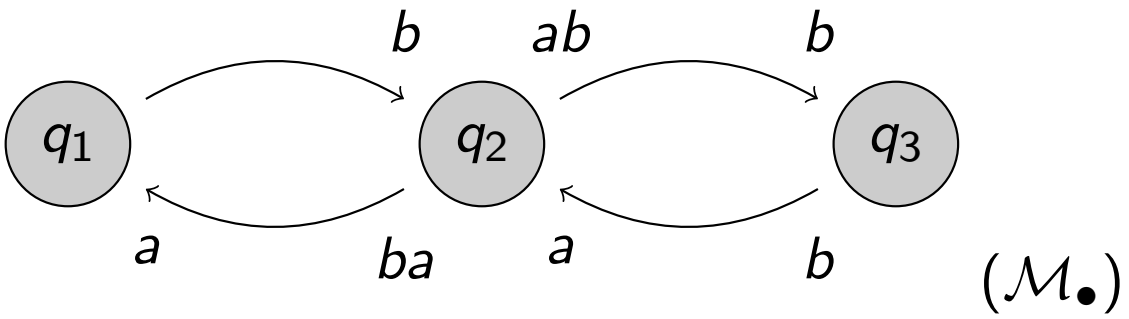
- Halts by taking 2 letters where there is 1.
- **Fact:** halting problem is undecidable

Example



$(\mathcal{M}_\bullet)$

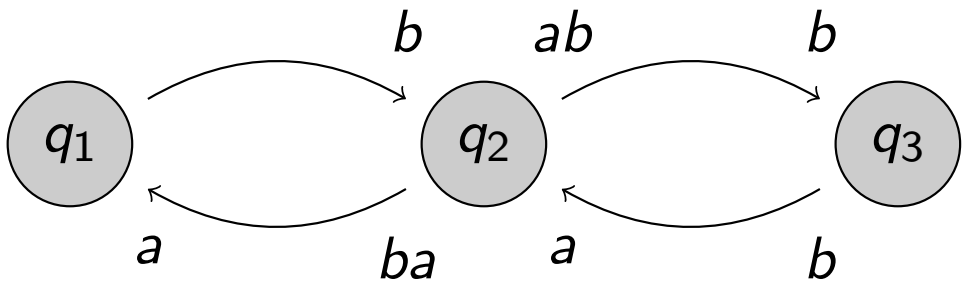
Example



$(\mathcal{M}_\bullet)$

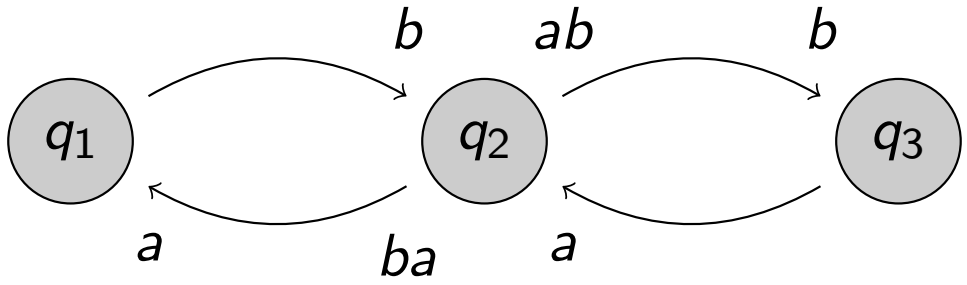
$$(q_1, a) \longrightarrow\!\triangleright_{\mathcal{M}_\bullet} (q_2, ab) \longrightarrow\!\triangleright_{\mathcal{M}_\bullet} (q_3, b) \longrightarrow\!\triangleright_{\mathcal{M}_\bullet} (q_2, a)$$

Example



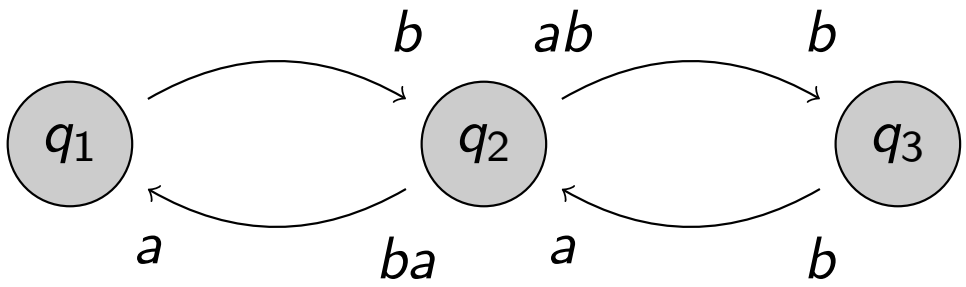
$(\mathcal{M}_\bullet)$

$$(q_1, a) \longrightarrow_{\mathcal{M}_\bullet} (q_2, ab) \longrightarrow_{\mathcal{M}_\bullet} (q_3, b) \longrightarrow_{\mathcal{M}_\bullet} (q_2, a)$$



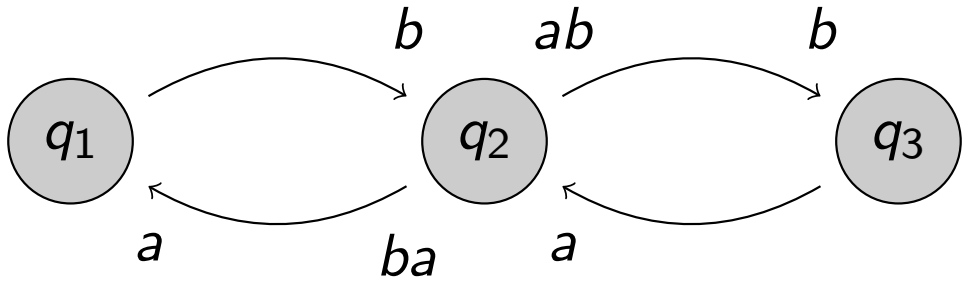
$(\mathcal{M}_{\circlearrowleft})$

Example



$(\mathcal{M}_\bullet)$

$$(q_1, a) \longrightarrow_{\mathcal{M}_\bullet} (q_2, ab) \longrightarrow_{\mathcal{M}_\bullet} (q_3, b) \longrightarrow_{\mathcal{M}_\bullet} (q_2, a)$$



$(\mathcal{M}_\circlearrowleft)$

$$(q_1, a) \longrightarrow_{\mathcal{M}_\circlearrowleft} (q_2, ab) \longrightarrow_{\mathcal{M}_\circlearrowleft} (q_3, b) \longrightarrow_{\mathcal{M}_\circlearrowleft} (q_2, ba) \longrightarrow_{\mathcal{M}_\circlearrowleft} (q_1, a)$$

Undecidability result

Take a **QM** $\langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$.

Undecidability result

Take a **QM** $\langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$.

Syntax Σ: constant $\mathtt{S}$, unary $\mathtt{q}$ for each $q \in Q$. and $\mathtt{a}$: for each $a \in A$

Undecidability result

Take a **QM** $\langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$.

Syntax Σ: constant $\mathtt{S}$, unary $\mathtt{q}$ for each $q \in Q$. and $\mathtt{a}$: for each $a \in A$

- $\mathtt{S} \longrightarrow \mathtt{a}_1 : \mathtt{q}_1(\mathtt{S})$ (**S**), where (q_1, a_1) is the initial configuration,

Undecidability result

Take a **QM** $\langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$.

Syntax Σ: constant $\mathtt{S}$, unary $\mathtt{q}$ for each $q \in Q$. and $\mathtt{a}$: for each $a \in A$

- $\mathtt{S} \longrightarrow \mathtt{a}_1 : \mathtt{q}_1(\mathtt{S})$ (**S**), where (q_1, a_1) is the initial configuration,
- $\mathtt{q}(x) \longrightarrow \mathtt{c} : \mathtt{q}'(x)$ (**R0**), whenever $\delta_0(q) = (c, q')$,

Undecidability result

Take a **QM** $\langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$.

Syntax Σ: constant $\mathtt{S}$, unary $\mathtt{q}$ for each $q \in Q$. and $\mathtt{a}$: for each $a \in A$

- $\mathtt{S} \longrightarrow \mathtt{a}_1 : \mathtt{q}_1(\mathtt{S})$ (**S**), where (q_1, a_1) is the initial configuration,
- $\mathtt{q}(x) \longrightarrow \mathtt{c} : \mathtt{q}'(x)$ (**R0**), whenever $\delta_0(q) = (c, q')$,
- $\mathtt{q}(\mathtt{a} : y) \longrightarrow \mathtt{c} : \mathtt{q}'(y)$ (**R1**), whenever $\delta_1(q, a) = (c, q')$,

Undecidability result

Take a **QM** $\langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$.

Syntax Σ: constant $\mathtt{S}$, unary $\mathtt{q}$ for each $q \in Q$. and $\mathtt{a}$: for each $a \in A$

- $\mathtt{S} \longrightarrow \mathtt{a}_1 : \mathtt{q}_1(\mathtt{S})$ (**S**), where (q_1, a_1) is the initial configuration,
- $\mathtt{q}(x) \longrightarrow \mathtt{c} : \mathtt{q}'(x)$ (**R0**), whenever $\delta_0(q) = (c, q')$,
- $\mathtt{q}(\mathtt{a} : y) \longrightarrow \mathtt{c} : \mathtt{q}'(y)$ (**R1**), whenever $\delta_1(q, a) = (c, q')$,
- $\mathtt{q}(\mathtt{a} : \mathtt{b} : z) \longrightarrow \mathtt{c} : \mathtt{q}'(z)$ (**R2**), whenever $\delta_2(q, a, b) = (c, q')$.

Undecidability result

Take a **QM** $\langle Q, A, a_1, q_1, \delta_0, \delta_1, \delta_2 \rangle$.

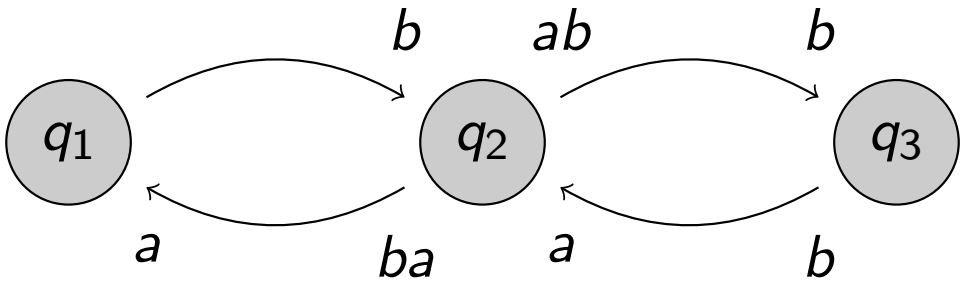
Syntax Σ: constant $\mathtt{S}$, unary $\mathtt{q}$ for each $q \in Q$. and $\mathtt{a}$: for each $a \in A$

- $\mathtt{S} \longrightarrow \mathtt{a}_1 : \mathtt{q}_1(\mathtt{S})$ (**S**), where (q_1, a_1) is the initial configuration,
- $\mathtt{q}(x) \longrightarrow \mathtt{c} : \mathtt{q}'(x)$ (**R0**), whenever $\delta_0(q) = (c, q')$,
- $\mathtt{q}(\mathtt{a} : y) \longrightarrow \mathtt{c} : \mathtt{q}'(y)$ (**R1**), whenever $\delta_1(q, a) = (c, q')$,
- $\mathtt{q}(\mathtt{a} : \mathtt{b} : z) \longrightarrow \mathtt{c} : \mathtt{q}'(z)$ (**R2**), whenever $\delta_2(q, a, b) = (c, q')$.

Theorem

$\mathcal{M}$ halts from the initial configuration iff $S_{\mathcal{M}}$ is not productive for $\mathtt{S}$.

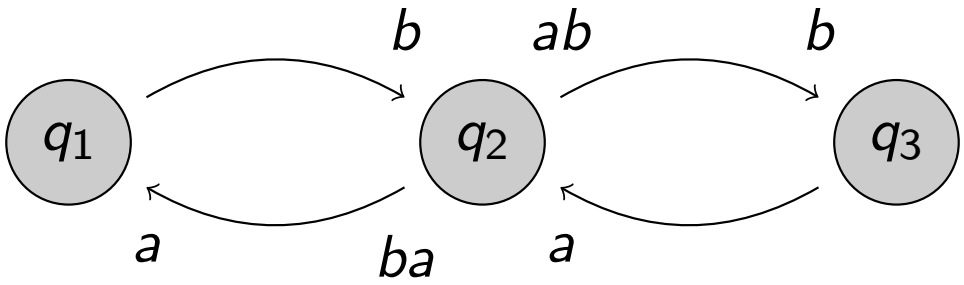
Halts = Not Productive



$$\mathbf{S}_{\mathcal{M}}. \quad \mathtt{S} \longrightarrow \mathtt{a} : \mathtt{q}_1(\mathtt{S})\ (\mathbf{1}) \qquad \mathtt{q}_1(x) \longrightarrow \mathtt{b} : \mathtt{q}_2(x)\ (\mathbf{2})$$
$$\mathtt{q}_3(\mathtt{b} : y) \longrightarrow \mathtt{a} : \mathtt{q}_2(y)\ (\mathbf{3}) \qquad \mathtt{q}_2(\mathtt{a} : \mathtt{b} : z) \longrightarrow \mathtt{b} : \mathtt{q}_3(z)\ (\mathbf{4})$$
$$\mathtt{q}_2(\mathtt{b} : \mathtt{a} : z) \longrightarrow \mathtt{a} : \mathtt{q}_1(z)(\mathbf{5})$$

$$\mathtt{S} \to \mathtt{a} : \mathtt{q}_1(\mathtt{S}) \to \mathtt{a} : \mathtt{b} : \mathtt{q}_2(\mathtt{S}) \to \cdots \to \mathtt{a} : \mathtt{b} : \mathtt{q}_2(\mathtt{a} : \mathtt{b} : \mathtt{q}_2(\mathtt{S})) \to$$
$$\mathtt{a} : \mathtt{b} : \mathtt{b} : \mathtt{q}_3(\mathtt{q}_2(\mathtt{S})) \to \cdots \to \mathtt{a} : \mathtt{b} : \mathtt{b} : \mathtt{a} : \mathtt{a} : \mathtt{q}_1(\mathtt{q}_3(\mathtt{S})) \to$$
$$\mathtt{a} : \mathtt{b} : \mathtt{b} : \mathtt{a} : \mathtt{a} : \mathtt{b} : \mathtt{q}_2(\mathtt{q}_3(\mathtt{S})) \to \mathtt{a} : \mathtt{b} : \mathtt{b} : \mathtt{a} : \mathtt{a} : \mathtt{b} : \mathtt{q}_2(\mathtt{q}_3(\mathtt{a} : \mathtt{q}_1(\mathtt{S})))$$

Loops = Productive



$$\mathbf{S}_{\mathcal{M}_{\circlearrowleft}} \qquad \mathtt{S} \longrightarrow \mathtt{a} : \mathtt{q}_1(\mathtt{S})\ (\mathbf{1}) \qquad \mathtt{q}_1(x) \longrightarrow \mathtt{b} : \mathtt{q}_2(x)\ (\mathbf{2})$$
$$\mathtt{q}_3(y) \longrightarrow \mathtt{a} : \mathtt{q}_2(y)\ (\mathbf{3'}) \qquad \mathtt{q}_2(\mathtt{a} : \mathtt{b} : z) \longrightarrow \mathtt{b} : \mathtt{q}_3(z)\ (\mathbf{4})$$
$$\mathtt{q}_2(\mathtt{b} : \mathtt{a} : z) \longrightarrow \mathtt{a} : \mathtt{q}_1(z)(\mathbf{5})$$

$\mathtt{S} \to \mathtt{a} : \mathtt{q}_1(\mathtt{S}) \to \cdots \to \mathtt{a} : \mathtt{b} : \mathtt{b} : \mathtt{a} : \mathtt{a} : \mathtt{b} : \mathtt{q}_2(\mathtt{q}_3(\mathtt{S})) \to \cdots$

$\mathtt{a} : \mathtt{b} : \mathtt{b} : \mathtt{a} : \mathtt{a} : \mathtt{b} : \mathtt{q}_2(\mathtt{a} : \mathtt{q}_2(\mathtt{a} : \mathtt{b} : \mathtt{q}_2(\mathtt{S}))) \to \mathtt{a} : \mathtt{b} : \mathtt{b} : \mathtt{a} : \mathtt{a} : \mathtt{b} : \mathtt{q}_2(\mathtt{a} : \mathtt{b} : \mathtt{q}_3(\mathtt{q}_2(\mathtt{S})))$

$\to \mathtt{a} : \mathtt{b} : \mathtt{b} : \mathtt{a} : \mathtt{a} : \mathtt{b} : \mathtt{b} : \mathtt{q}_3(\mathtt{q}_3(\mathtt{q}_2(\mathtt{S}))) \to \cdots$

Data-oblivious Pure Stream Specification

$$\begin{aligned} \mathtt{S} &\to \mathtt{q}(0:1:\mathtt{S}) \\ \mathtt{q}(0:\sigma) &\to 0:1:\mathtt{q}(\sigma) \\ \mathtt{q}(1:\sigma) &\to 1:0:q(\sigma) \end{aligned}$$

$$\Longrightarrow$$

$$\begin{aligned} \mathtt{S} &\to \mathtt{q}(\bullet:\bullet:\mathtt{S}) \\ \mathtt{q}(\bullet:\sigma) &\to \bullet:\bullet:\mathtt{q}(\sigma) \\ \mathtt{q}(\bullet:\sigma) &\to \bullet:\bullet:q(\sigma) \end{aligned}$$

Data-oblivious Pure Stream Specification

$$\begin{array}{c} \mathtt{S} \to \mathtt{q}(0:1:\mathtt{S}) \\ \mathtt{q}(0:\sigma) \to 0:1:\mathtt{q}(\sigma) \\ \mathtt{q}(1:\sigma) \to 1:0:q(\sigma) \end{array} \quad \Longrightarrow \quad \begin{array}{c} \mathtt{S} \to \mathtt{q}(\bullet:\bullet:\mathtt{S}) \\ \mathtt{q}(\bullet:\sigma) \to \bullet:\bullet:\mathtt{q}(\sigma) \\ \mathtt{q}(\bullet:\sigma) \to \bullet:\bullet:q(\sigma) \end{array}$$

Non data-oblivious Specification

$$\begin{array}{c} \mathtt{S} \longrightarrow \mathtt{a}:\mathtt{q}_1(\mathtt{S}) \\ \mathtt{q}_1(x) \longrightarrow \mathtt{b}:\mathtt{q}_2(x) \\ \mathtt{q}_3(\mathtt{b}:y) \longrightarrow \mathtt{a}:\mathtt{q}_2(y) \\ \mathtt{q}_2(\mathtt{a}:\mathtt{b}:z) \longrightarrow \mathtt{b}:\mathtt{q}_3(z) \\ \mathtt{q}_2(\mathtt{b}:\mathtt{a}:z) \longrightarrow \mathtt{a}:\mathtt{q}_1(z) \end{array} \quad \Longrightarrow \quad \begin{array}{c} \mathtt{S} \longrightarrow \bullet:\mathtt{q}_1(\mathtt{S}) \\ \mathtt{q}_1(x) \longrightarrow \bullet:\mathtt{q}_2(x) \\ \mathtt{q}_3(\bullet:y) \longrightarrow \bullet:\mathtt{q}_2(y) \\ \mathtt{q}_2(\bullet:\bullet:z) \longrightarrow \bullet:\mathtt{q}_3(z) \\ \mathtt{q}_2(\bullet:\bullet:z) \longrightarrow \bullet:\mathtt{q}_1(z) \end{array}$$