Pushdown Automata and Context-Free Grammars in Bisimulation Semantics

Jos Baeten Cesare Carissimo Bas Luttik

CWI, Amsterdam

University of Amsterdam

Eindhoven University of Technology

CALCO, 2 September 2021

Well-known theorem

A language can be defined by a pushdown automaton iff it can be defined by a context-free grammar.

Well-known theorem

A language can be defined by a pushdown automaton iff it can be defined by a context-free grammar.

A process can be defined by a pushdown automaton iff it can be defined by a finite guarded sequential recursive specification, with a notion of state awareness added.

Definition

A *language* is a language equivalence class of process graphs.

A *process* is a bisimulation equivalence class of process graphs.

CWI

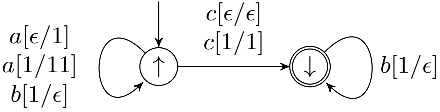
Definition

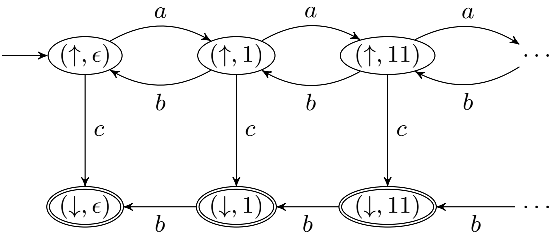
A *language* is a language equivalence class of process graphs.

A *process* is a bisimulation equivalence class of process graphs.

A *process graph* is a non-deterministic automaton, possibly infinite.
A *process graph* is a labelled transition system with an initial state.

Pushdown Automaton





Context-Free Processes

- Use SOS to give automata for syntax $\mathbf{0}, \mathbf{1}, a., ;, +$
- (Used this to tackle the theorem since CONCUR 2008)

$$\frac{}{\mathbf{1}\downarrow} \qquad \frac{}{a.p \xrightarrow{a} p}$$

$$\frac{p \xrightarrow{a} p'}{(p+q) \xrightarrow{a} p'} \qquad \frac{q \xrightarrow{a} q'}{(p+q) \xrightarrow{a} q'} \qquad \frac{p\downarrow}{(p+q)\downarrow} \qquad \frac{q\downarrow}{(p+q)\downarrow}$$

$$\frac{p \xrightarrow{a} p'}{p\,;q \xrightarrow{a} p'\,;q} \qquad \frac{p\downarrow \quad q \xrightarrow{a} q'}{p\,;q \xrightarrow{a} q'} \qquad \frac{p\downarrow \quad q\downarrow}{p\,;q\downarrow}$$

Context-Free Processes

- Use SOS to give automata for syntax $\mathbf{0}, \mathbf{1}, a., ;, +$
- (Together with MSc student Astrid Belder)

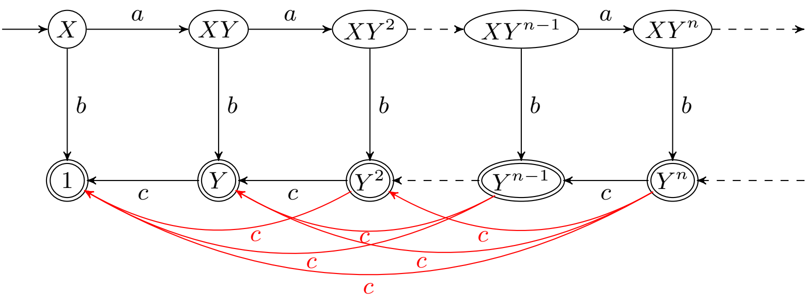
$$\frac{}{\mathbf{1}\downarrow} \qquad \frac{}{a.p \xrightarrow{a} p}$$

$$\frac{p \xrightarrow{a} p'}{(p+q) \xrightarrow{a} p'} \qquad \frac{q \xrightarrow{a} q'}{(p+q) \xrightarrow{a} q'} \qquad \frac{p\downarrow}{(p+q)\downarrow} \qquad \frac{q\downarrow}{(p+q)\downarrow}$$

$$\frac{p \xrightarrow{a} p'}{p\,;q \xrightarrow{a} p'\,;q} \qquad \frac{p\downarrow \quad p \not\rightarrow \quad q \xrightarrow{a} q'}{p\,;q \xrightarrow{a} q'} \qquad \frac{p\downarrow \quad q\downarrow}{p\,;q\downarrow}$$

The difference

$$X \stackrel{\text{def}}{=} a.(X\,;Y) + b.\mathbf{1} \qquad Y \stackrel{\text{def}}{=} c.\mathbf{1} + \mathbf{1} \quad .$$



Recursion

$$\frac{p \xrightarrow{a} p' \quad (N = p) \in E}{N \xrightarrow{a} p'} \qquad \frac{p\downarrow \quad (N = p) \in E}{N\downarrow}$$

Limit to finite *guarded* recursive specifications.
Greibach normal form $X = (\mathbf{1}+) \sum_{i=1}^{n} a_i.\xi_i$.

CWI

Bisimulation

$p \leftrightarrows q$, p is *bisimilar* to q if there is a symmetric binary relation R with $p \; R \; q$ satisfying the following conditions:

1. whenever $s \; R \; t$ and $s \xrightarrow{a} s'$, there is t' such that $t \xrightarrow{a} t'$ and $s' \; R \; t'$; and
2. whenever $s \; R \; t$ and $s\downarrow$, then $t\downarrow$.

Context-free Grammar

A recursive specification for the process of $\{a^n b^n \mid n \geq 0\}$ is

$$X = \mathbf{1} + a.Y$$

$$Y = b.\mathbf{1} + a.Y; b.\mathbf{1}$$

Context-free Grammar

A recursive specification for the process of $\{a^n b^n \mid n \geq 0\}$ is

$$X = \mathbf{1} + a.Y$$

$$Y = b.\mathbf{1} + a.Y\,; b.\mathbf{1}$$

A recursive specification for the always accepting stack is

$$S = \mathbf{1} + \sum_{d \in D} push(d).T_d\,; S$$

$$T_d = \mathbf{1} + pop(d).\mathbf{1} + \sum_{e \in D} push(e).T_e\,; T_d$$

CWI

Theorem 1

For every guarded sequential specification there is a pushdown automaton with the same process (with two non-bisimilar states).

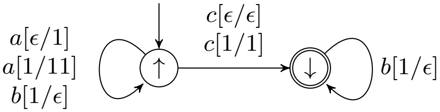
Theorem 2

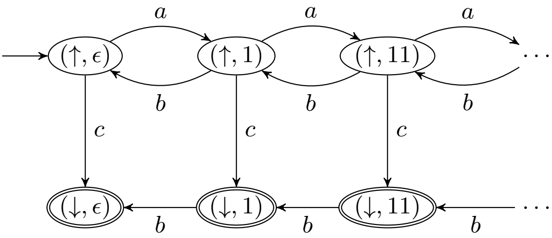
For every one-state pushdown automaton there is a guarded sequential specification with the same process.

Theorem 3

There is a pushdown automaton with two states, such that there is no guarded sequential specification with the same process.

Pushdown Automaton



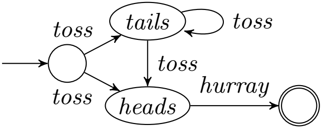


Signals and conditions

- The visible part of the state of a process is a proposition, an expression in propositional logic
- $P_1, \ldots, P_n$ propositional variables, constants $true, false$, logical connectives
- $\phi \, {}^{\wedge}\!\blacktriangle \, x$ is root signal emission
- $\phi :\rightarrow x$ is guarded command
- Comes with a valuation in every state of the transition system (BBergstra 1997)
- Stateless bisimulation

CWI

Example: coin toss



$$T \stackrel{\text{def}}{=} toss.(heads \wedge\!\!\blacktriangle \mathbf{1}) + toss.(tails \wedge\!\!\blacktriangle \mathbf{1})$$

$$S \stackrel{\text{def}}{=} T\,;(heads :\rightarrow hurray.\mathbf{1} + tails :\rightarrow S)$$

Theorem 4

For every pushdown automaton there is a guarded sequential specification with signals and conditions with the same process.

$$S = a.(state{\uparrow} \,{}^{\wedge}\!\!\blacktriangle A\,; (state{\uparrow} :\rightarrow S + state{\downarrow} :\rightarrow \mathbf{1})) + c.(state{\downarrow} \,{}^{\wedge}\!\!\blacktriangle \mathbf{1})$$

$$A = state{\downarrow} :\rightarrow b.(state{\downarrow} \,{}^{\wedge}\!\!\blacktriangle \mathbf{1}) +$$

$$+\, state{\uparrow} :\rightarrow (a.(state{\uparrow} \,{}^{\wedge}\!\!\blacktriangle A; A) + b.(state{\uparrow} \,{}^{\wedge}\!\!\blacktriangle \mathbf{1}) + c.(state{\downarrow} \,{}^{\wedge}\!\!\blacktriangle A)).$$

Theorem 5

For every guarded sequential specification with signals and conditions there is a pushdown automaton with the same process.

Conclusion

Interaction is a key ingredient of any computer.
A model of computation needs to incorporate interaction.
Aim is a full integration of automata theory and process theory.
Result is a richer and more refined theory.
Turn lecture notes into a text book.