Strictness of the Collapsible Pushdown Graph Hierarchy

Alexander Kartzow (joint work with Pawel Parys)

Universität Leipzig

27.08.2012

supported by DFG project GELO and ESF project GAMES

Collapsible Pushdown Systems (CPS)

- Higher-order pushdown systems (HOPS) [Maslov'76]
  - Pushdown systems with nested stack of ... of stacks
  - Operation: push / pop for each stack level

- Theorem (Knapik, Niwinski, Urzyczyn '02)

  trees of HOPS = *trees of safe higher-order recursion schemes*

Collapsible Pushdown Systems (CPS)

- Higher-order pushdown systems (HOPS) [Maslov'76]
  - Pushdown systems with nested stack of ... of stacks
  - Operation: push / pop for each stack level
- Collapsible pushdown system (CPS)
  Extension by "Collapse" operation
  defined by Hague, Murawski, Ong and Serre in '08

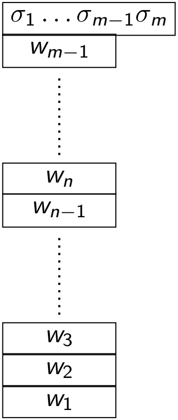
- Theorem (Knapik, Niwinski, Urzyczyn '02)
  trees of HOPS = trees of safe higher-order recursion schemes

  Theorem (Hague et al. '08)
  trees of CPS = trees of higher-order recursion schemes

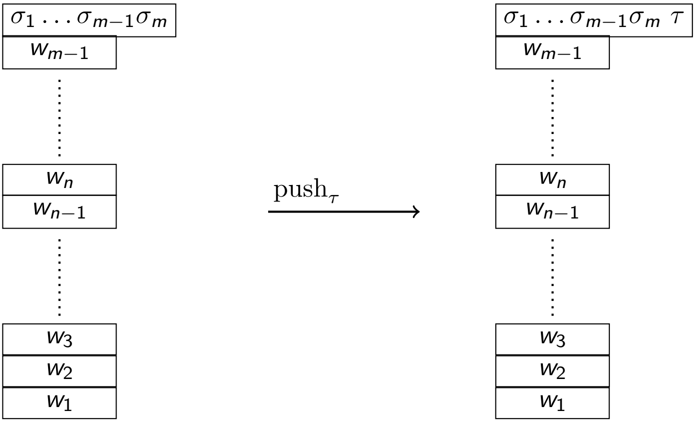
Proper-Hierarchy-Questions

- Are there more level $i + 1$ graphs than level i?
- Are there more level $i + 1$ trees than level i?
- Are there more languages in level $i + 1$ than in level i?
- Does the collapse operation make a difference?

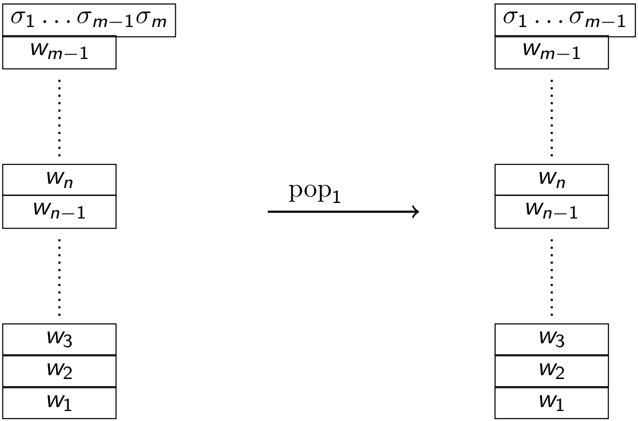
Stack Operations



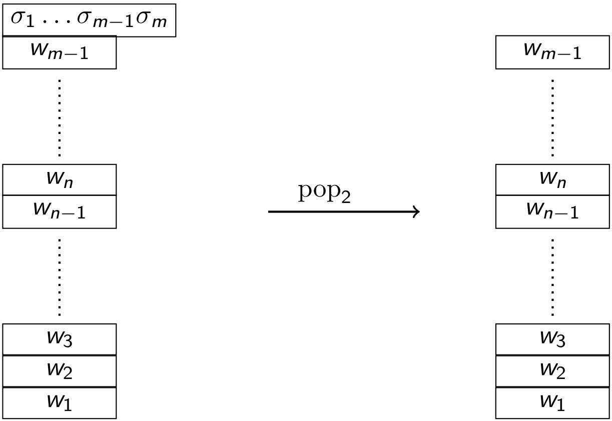
Stack Operations



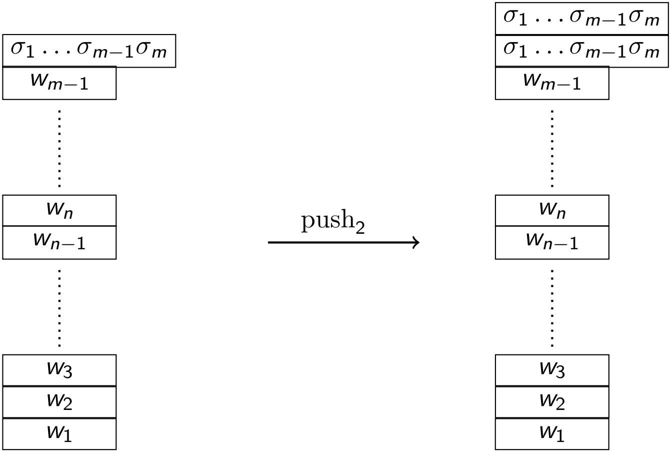
Stack Operations



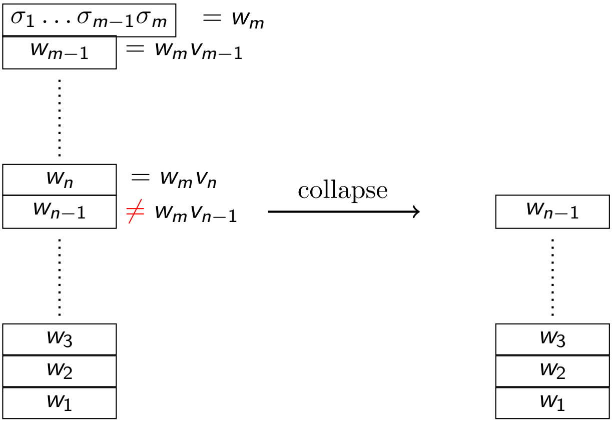
Stack Operations



Stack Operations



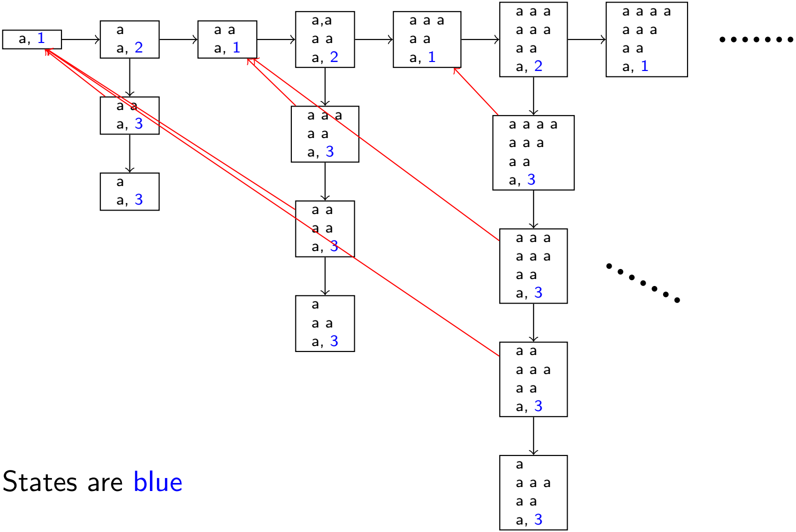
Stack Operations



Definition Collapsible Pushdown Graph

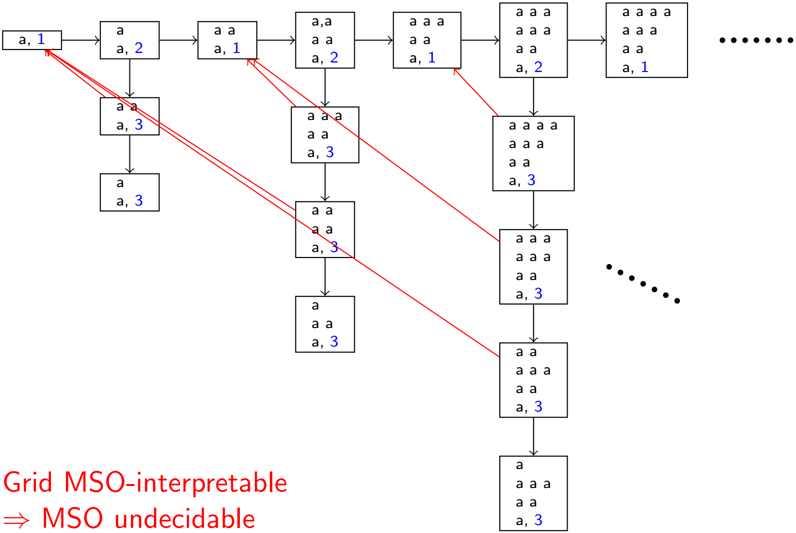
- Transition relation Δ:
  state $+$ topmost letter $\mapsto$ new state $+$ stack-operation
  e.g. $\delta = (q, \sigma) \mapsto (q', \mathrm{pop}_2)$
- Configuration (q, s) – q state, s stack (of level 2)
- $(q, s) \xrightarrow{\delta} (q', \mathrm{pop}_2(s))$ **if** top of s is σ
- CPG: configurations of CPS $+$ labelled transition relation
- CPG$/\varepsilon$: ε-contraction of CPG

Example of CPG



States are blue

Example of CPG



Grid MSO-interpretable

$\Rightarrow$ MSO undecidable

CPS as Counting Machines

Definition

$f : \mathbb{N} \to \mathbb{N}$ a function

A deterministic CPS $\mathcal{S}$ is an *f-countdown* iff

$\mathcal{S}$ started in (q_0, a^n) makes exactly $f(n)$ non-ε computation steps.

Theorem

For $f_k(x) := \exp_{k-1}(x)$, there is an f_k-coundown of level k.

Proof.

Level 1: $f_0(x) = \exp_0(x) = x$ $(q_0, a, \gamma, \mathrm{pop}, q_0)$

Level 2: 1-stacks = exponents

$$\boxed{\begin{array}{l} \mathtt{a\ a\ a} \\ \mathtt{a\ a\ a\ a\ a} \end{array}}$$

$$\left.\begin{array}{l} 2^3 = 8 \\ 2^5 = 32 \end{array}\right\} = 40$$

$\square$

CPS as Counting Machines

Definition

$f : \mathbb{N} \to \mathbb{N}$ a function

A deterministic CPS $\mathcal{S}$ is an *f-countdown* iff

$\mathcal{S}$ started in (q_0, a^n) makes exactly $f(n)$ non-ε computation steps.

Theorem

For $f_k(x) := \exp_{k-1}(x)$, *there is an* f_k*-coundown of level* k*.*

Proof.

Level 1: $f_0(x) = \exp_0(x) = x$ $\quad (q_0, a, \gamma, \mathrm{pop}, q_0)$

Level 2: 1-stacks = exponents $\quad 2^k - 1 = 2^{k-1} + 2^{k-2} + \cdots + 2^0$

$$\boxed{\begin{array}{l} \mathtt{a\ a\ a} \\ \mathtt{a\ a\ a\ a\ a} \end{array}} \xrightarrow{(\mathrm{pop}_1 \mathrm{push}_2)^*} \boxed{\begin{array}{l} \mathtt{a} \\ \mathtt{a\ a} \\ \mathtt{a\ a\ a\ a\ a} \end{array}}$$

$$\left.\begin{array}{l} 2^3 = 8 \\ 2^5 = 32 \end{array}\right\} = 40$$

$$\left.\begin{array}{l} 2^0 = 1 \\ 2^1 = 2 \\ 2^2 = 4 \\ 2^5 = 32 \end{array}\right\} = 39$$

CPS as Counting Machines

Definition

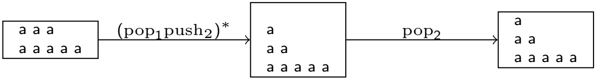
$f : \mathbb{N} \to \mathbb{N}$ a function
A deterministic CPS $\mathcal{S}$ is an *f-countdown* iff
$\mathcal{S}$ started in (q_0, a^n) makes exactly $f(n)$ non-ε computation steps.

Theorem

For $f_k(x) := \exp_{k-1}(x)$, there is an f_k-coundown of level k.

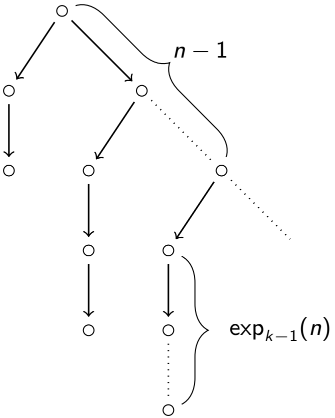
Proof.

Level 1: $f_0(x) = \exp_0(x) = x \qquad (q_0, a, \gamma, \mathrm{pop}, q_0)$
Level 2: 1-stacks = exponents $\qquad 2^k - 1 = 2^{k-1} + 2^{k-2} + \cdots + 2^0$



$$\left.\begin{array}{l} 2^3 = 8 \\ 2^5 = 32 \end{array}\right\} = 40 \qquad \left.\begin{array}{l} 2^0 = 1 \\ 2^1 = 2 \\ 2^2 = 4 \\ 2^5 = 32 \end{array}\right\} = 39 \qquad \left.\begin{array}{l} 2^1 = 2 \\ 2^2 = 4 \\ 2^5 = 32 \end{array}\right\} = 38$$

Another Example of CPG



Example

$\mathfrak{T} := (T, \mathrm{succ})$ with $T := \{0\}^* \cup \{0^{n-1}1^j : 0 \leq j \leq \exp_{k-1}(n)\}$ is a k-CPG$/\varepsilon$

The Pumping Lemma

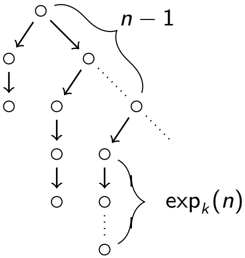
Theorem

$\mathfrak{G}$ *k-CPG/ε finitely branching*
$\exists C \in \mathbb{N}$ *: for $g_0 \in \mathfrak{G}$ at distance n from the initial configuration*
$\exists g_1 \quad \mathrm{dist}(g_0, g_1) = \exp_{k-1}(C \cdot (n+1))$
$\Rightarrow$ *Infinitely many paths start at g_0.*

Corollary

The collapsible pushdown graph hierarchy is strict level-by-level.
The collapsible pushdown tree hierarchy is strict level-by-level.

Application



Example

$\mathfrak{T} := (T, \mathrm{succ})$ with
$T := \{0\}^* \cup \{0^{n-1}1^j : 0 \leq j \leq \exp_k(n)\}$
is not a k-CPG$/\varepsilon$,
but is a $(k+1)$-CPG$/\varepsilon$.

Proof.

Choose $2^{n_0} > C \cdot (n_0 + 1)$ then
$\exp_k(n_0) = \exp_{k-1}(2^{n_0}) > \exp_{k-1}(C \cdot (n_0 + 1))$
$\stackrel{\text{P.L.}}{\Rightarrow}$ infinitely many paths start at $0^{n_0-1}1$ contradiction $\square$

Pumpable Runs

Definition (Increasing Run in 1-PS)

initial stack is prefix of all stacks in the run

$$R_1 : q_1, aa \quad \to q_2, aab \quad \to q_3, aa \quad \to q_4, aab \quad \to q_1, aaba$$

Examples

R_1 is an increasing run

Pumpable Runs

Definition (Increasing Run in 1-PS)

initial stack is prefix of all stacks in the run

$$
\begin{array}{llllll}
R_2 : q_1, aa & \rightarrow q_2, a & \rightarrow q_3, aa & \rightarrow q_4, aab & \rightarrow q_1, aaba \\
R_1 : q_1, aa & \rightarrow q_2, aab & \rightarrow q_3, aa & \rightarrow q_4, aab & \rightarrow q_1, aaba
\end{array}
$$

Examples

R_1 is an increasing run

R_2 is not an increasing run

Pumpable Runs

Definition (Increasing Run in 1-PS)

initial stack is prefix of all stacks in the run

$$
\begin{array}{llllll}
R_2 : q_1, aa & \to q_2, {\color{red}a} & \to q_3, aa & \to q_4, aab & \to q_1, aaba \\
R_1 : q_1, aa & \to q_2, aab & \to q_3, aa & \to q_4, aab & \to q_1, aaba \\
 & \to q_2, aabaab & \to q_3, aabaa & \to q_4, aabaab & \to q_1, aabaaba
\end{array}
$$

Examples

R_1 is an increasing run

R_2 is not an increasing run

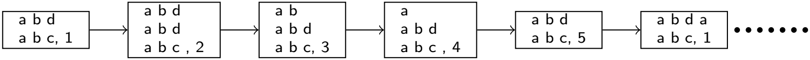
Increasing run with

- initial state = final state
- initial top symbol = final top symbol

is *pumpable*.

Increasing Runs on Higher Levels

Example



Proof of the pumping lemma:

- Describe increasing runs with *context free run grammar*
  nonterminals = sets of runs; terminals =transitions
  Example: $Q \supseteq \delta Q | \varepsilon$
- Context free run grammar induces *type* function on configurations
  type : Stacks $\to D$, D a finite set such that
  $(q, s) \to^* (q', s')$ increasing run and $\text{type}(s) = \text{type}(t)$
  $\Rightarrow \exists t' \quad \text{type}(s') = \text{type}(t')$ and $(q, t) \to^* (q', t')$ increasing run
- Combinatorics: long run contains many increasing runs
  $\Rightarrow \exists$ increasing run with equal initial and final type.

More Applications of Grammars / Types

Theorem

Given $\mathfrak{G}$ a $k - CPG/\varepsilon$, it is decidable (in $\exp_{O(n)}$-time) whether

- *$\mathfrak{G}$ is finitely branching*
- *$\mathfrak{G}$ contains a loop*
- *$\mathfrak{G}$ is finite*
- *the unfolding of $\mathfrak{G}$ into a tree is finite*

Proof idea

1. $\exists C$ property holds iff a run in class C exists
2. provide context-free run grammar G for C
3. Check the type (w.r.t G) of the initial configuration

More Applications of Grammars / Types

Theorem

Given $\mathfrak{G}$ a $k - CPG/\varepsilon$, it is decidable (in $\exp_{O(n)}$-time) whether

- *$\mathfrak{G}$ is finitely branching*
- *$\mathfrak{G}$ contains a loop*
- *$\mathfrak{G}$ is finite*
- *the unfolding of $\mathfrak{G}$ into a tree is finite*

Theorem (Parys '12)

Collapse operation increases the power of higher-order pushdowns

- *More configuration graphs with collapse*
- *More trees with collapse*
  $\Rightarrow$ *Safety restricts recursion schemes*
- *More languages accepted with collapse*

Conclusion and Open Problems

Conclusion

- pumping lemma for k-CPG$/\varepsilon$: tool for disproving membership
- $\Rightarrow$ strictness (level-by-level) of the CPG hierarchy
- Proof strategy also yields decidability of
  - finite branching
  - finiteness
  - loop-freeness
  - finiteness of unfolding

Open questions

- Level-by-level separation of languages accepted by k-CPS
- Stronger pumping: more information about the resulting paths