Pointers and Polynomial Space Functions

Isabel Oitavem

UNL and CMAF
Overview

- Implicit characterizations of classes of computational complexity: a uniform approach.
- Recursion schemes over algebras.
- Implicit characterization of $Pspace$. 
Uniform approach

Given a free algebra $\mathcal{A}$ define

$T_\mathcal{A} = \text{COMP}/\text{REC}_\mathcal{A}\{\mathcal{A} \text{ constructors, destructors, conditional, proj.}\}$

$ST_\mathcal{A} = \text{SCOMP}/\text{SREC}_\mathcal{A}\{\mathcal{A} \text{ constr, destr., cond., proj. in both tiers}\}$

$ST_N = Lspace$

$ST_W = Ptime$

$ST_T = NC$
Uniform approach — recursion schemes

<table>
<thead>
<tr>
<th>algebras</th>
<th>constructors</th>
<th>arities</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>$\epsilon, S_0, S_1$</td>
<td>0, 1, 1</td>
</tr>
<tr>
<td>T</td>
<td>$0, 1, \ast$</td>
<td>0, 0, 2</td>
</tr>
</tbody>
</table>

\[
\text{REC}_W: \quad 
\begin{align*}
  f(\epsilon, x) &= g(\epsilon, x) \\
  f(S_0z, x) &= h(S_0z, x, f(z, x)) \\
  f(S_1z, x) &= h(S_1z, x, f(z, x))
\end{align*}
\]

\[
\text{REC}_T: \quad 
\begin{align*}
  f(p, 0, x) &= g(p, 0, x) \\
  f(p, 1, x) &= g(p, 1, x) \\
  f(p, u * v, x) &= h(p, u * v, x, f(p * 0, u, x), f(p * 1, v, x))
\end{align*}
\]
Recursion schemes — pointers

**REC\textsubscript{W}:**

\[ f(\varepsilon, \bar{x}) = g(\varepsilon, \bar{x}) \]
\[ f(S_0z, \bar{x}) = h(S_0z, \bar{x}, f(z, \bar{x})) \]
\[ f(S_1z, \bar{x}) = h(S_1z, \bar{x}, f(z, \bar{x})) \]

**REC\textsubscript{T}:**

\[ f(p, 0, \bar{x}) = g(p, 0, \bar{x}) \]
\[ f(p, 1, \bar{x}) = g(p, 1, \bar{x}) \]
\[ f(p, u \ast v, \bar{x}) = h(p, u \ast v, \bar{x}, f(p \ast 0, u, \bar{x}), f(p \ast 1, v, \bar{x})) \]

**REC\textsubscript{TW}:**

\[ f(p, \varepsilon, \bar{x}) = g(p, \varepsilon, \bar{x}) \]
\[ f(p, S_0z, \bar{x}) = h(p, S_0z, \bar{x}, f(S_0p, z, \bar{x}), f(S_1p, z, \bar{x})) \]
\[ f(p, S_1z, \bar{x}) = h(p, S_1z, \bar{x}, f(S_0p, z, \bar{x}), f(S_1p, z, \bar{x})) \]
Pointers and Pspace

SREC\textsubscript{TW}:

\begin{align*}
  f(p, \epsilon, \bar{x}; \bar{y}) &= g(p, \epsilon, \bar{x}; \bar{y}) \\
  f(p, S_0z, \bar{x}; \bar{y}) &= h(p, S_0z, \bar{x}; \bar{y}, f(S_0p, z, \bar{x}; \bar{y}), f(S_1p, z, \bar{x}; \bar{y})) \\
  f(p, S_1z, \bar{x}; \bar{y}) &= h(p, S_1z, \bar{x}; \bar{y}, f(S_0p, z, \bar{x}; \bar{y}), f(S_1p, z, \bar{x}; \bar{y}))
\end{align*}

ST\textsubscript{TW} = \text{SCOMP}/\text{SREC}\textsubscript{TW}\{\mathbb{W} \text{ constr, destr., cond., proj. in both tiers}\}

\[
\text{ST}_{\text{TW}} = \text{Pspace}
\]

Let \(f\) be a function over \(\mathbb{W}\). \(f\) is computable in polynomial space if, and only if, \(f\) is bitwise computable by an ATM in polynomial time, and \(|f(w)|\) is polynomial in \(|w|\).
Pointers and Pspace

We assume that non-terminating configurations have universal or existential states. To see if a ATM accepts an input $x$, we define a bottom-up labeling of its computation tree (or part of it) by the following rules: 1) the accepting leaves are labeled 1; 2) any existential node is labeled 1 if at least one of its sons has been labeled 1; 3) any universal node is labeled 1 if all its sons are labeled 1. The machine accepts the input if, and only if, the root is labeled 1.
Pointers and Pspace

Notice that if $f(\epsilon, 11;)$ is defined by recursion (with pointers) on its second input based on $g$ and $h$, then one has the term $h(\epsilon; h(0; g(00; ), g(01; )), h(1; g(10; ), g(11; )))$ (some inputs are omitted), which corresponds to the tree

```
  h_\epsilon
     \ /
    /  \\
   h0  h1
     \ /
    /  \\
   g00 g01 g10 g11
```

and it is suitable to carry out the bottom-up labeling described above (assuming that non-terminating configurations have two successor configurations).