Decision Problems for Additive Regular Functions

Rajeev Alur Mukund Raghothaman

University of Pennsylvania

Friday 12th July, 2013

What are we studying?

Regular functions

What are we studying?

Regular functions

Languages, $\Sigma^* \rightarrow \texttt{bool}$

What are we studying?

Regular functions

Languages, $\Sigma^* \rightarrow \texttt{bool}$ DFA

What are we studying?

Regular functions

Languages, $\Sigma^* \to \texttt{bool}$	DFA
String transductions, $\Sigma^* \to \Gamma^*$	SST

What are we studying?

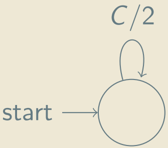
Regular functions from Σ^* to integers $\mathbb{Z}$

Languages, $\Sigma^* \to \texttt{bool}$	DFA
String transductions, $\Sigma^* \to \Gamma^*$	SST
Numerical functions, $\Sigma^* \to \mathbb{Z}$	?

Regular Functions

Modelling a coffee shop: Attempt 1

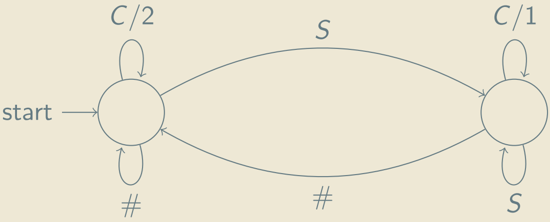
Finite automata with cost labels, a la Mealy machines



Regular Functions

Modelling a coffee shop: Attempt 1

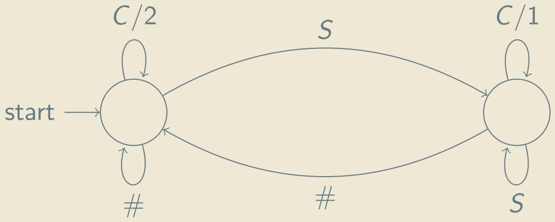
Finite automata with cost labels, a la Mealy machines



Regular Functions

Modelling a coffee shop: Attempt 1

Finite automata with cost labels, a la Mealy machines

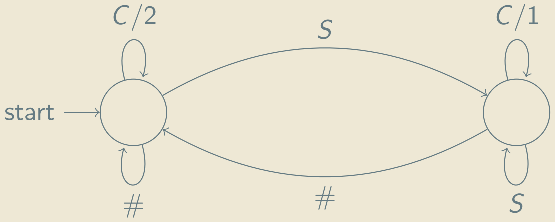


- Intuitive, analyzable

Regular Functions

Modelling a coffee shop: Attempt 1

Finite automata with cost labels, a la Mealy machines



- Intuitive, analyzable
- But not very expressive...

Regular Functions

Modelling a coffee shop: Attempt 1

What if the survey gives us a discount for coffee already purchased?

- Not possible if costs are paid up front
- Cost of an event cannot be influenced by later events

Regular Functions

Modelling a coffee shop: Attempt 1

What if the survey gives us a discount for coffee already purchased?

- Not possible if costs are paid up front
- Cost of an event cannot be influenced by later events

Solution?

Regular Functions

Modelling a coffee shop: Attempt 1

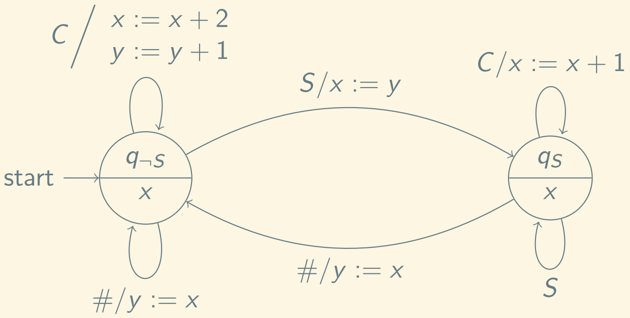
What if the survey gives us a discount for coffee already purchased?

- Not possible if costs are paid up front
- Cost of an event cannot be influenced by later events

Solution? Registers!

Regular Functions / Cost Register Automata

Modelling a coffee shop: Attempt 2



Regular Functions / Cost Register Automata

Properties, or why they're interesting

- Closure under linear combination, input reversal, etc.

- Fast equivalence procedure, decidable containment
- Equivalent to regular string-to-expression-tree transducers

Regular Functions / Cost Register Automata

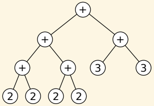
Properties, or why they're interesting

- Closure under linear combination, input reversal, etc.
  f^{rev} defined as $f^{rev}(\sigma) = f(\sigma^{rev})$ is regular when f is
- Fast equivalence procedure, decidable containment
- Equivalent to regular string-to-expression-tree transducers

Regular Functions / Cost Register Automata

Properties, or why they're interesting

- Closure under linear combination, input reversal, etc.
  f^{rev} defined as $f^{rev}(\sigma) = f(\sigma^{rev})$ is regular when f is
- Fast equivalence procedure, decidable containment
- Equivalent to regular string-to-expression-tree transducers

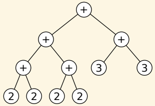
$abbbaaa\dots bba \longrightarrow$ 

Regular Functions / Cost Register Automata

Properties, or why they're interesting

- Closure under linear combination, input reversal, etc.
  f^{rev} defined as $f^{rev}(\sigma) = f(\sigma^{rev})$ is regular when f is
- Fast equivalence procedure, decidable containment
- Equivalent to regular string-to-expression-tree transducers

$$abbbaaa \ldots bba \longrightarrow$$



- Connections to weighted automata

What are we studying?

Regular functions from Σ^* to integers $\mathbb{Z}$

Languages, $\Sigma^* \to \texttt{bool}$	DFA
String transductions, $\Sigma^* \to \Gamma^*$	SST
Numerical functions, $\Sigma^* \to \mathbb{Z}$	?

What are we studying?

Cost register automata

Regular functions from Σ^* to integers $\mathbb{Z}$

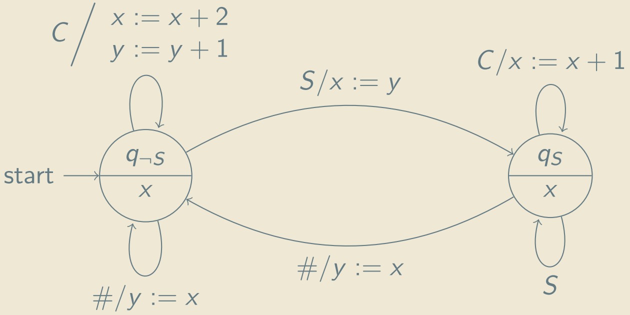
Languages, $\Sigma^* \to \texttt{bool}$	DFA
String transductions, $\Sigma^* \to \Gamma^*$	SST
Numerical functions, $\Sigma^* \to \mathbb{Z}$	CRA

Motivating Question: How do we Compute the Register Complexity?

Motivating Question

Register complexity

Does the coffee shop CRA really need 2 registers?



Register Separation

Register Separation

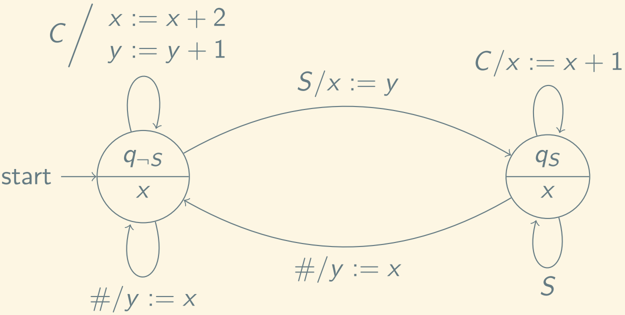
Confessions of a coffee addict

- Pick a large number, say $c = 1,000,000$

Register Separation

Confessions of a coffee addict

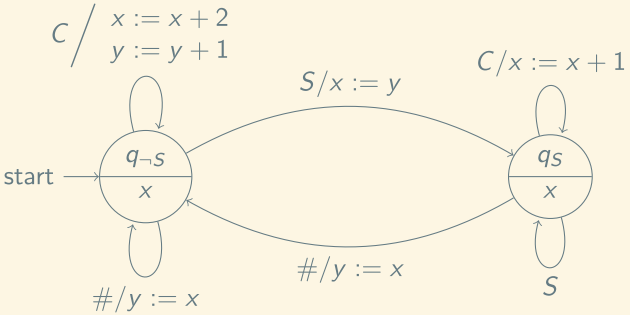
- Pick a large number, say $c = 1,000,000$
- Observe what happens after processing $C^c = CC \dots C$



Register Separation

Confessions of a coffee addict

- Pick a large number, say $c = 1,000,000$
- Observe what happens after processing $C^c = CC \ldots C$
- $|x - y| \geq c$



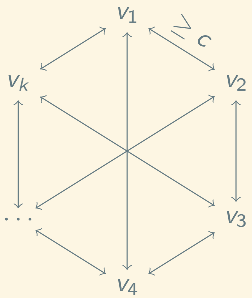
Register Separation

- In general, for each c, there is a path to $q_{\neg S}$ so $|x - y| \geq c$
- No 1-register machine can make up these arbitrary differences in finite time

Register Separation

Generalizing to k registers

- Pick a state q, and k registers
- Say, for each c, there is a string to q so every pair is at least c apart



- Then k registers are really necessary

Register Separation

Establishing the Converse

Register Separation

Establishing the converse

Claim

If the registers are not k-separable, then $k-1$ registers suffice

Register Separation

Establishing the converse

Claim

If the registers are not k-separable, then $k-1$ registers suffice

Separation: $\exists q,\ \forall c,\ \exists \sigma,\ \forall u, v,\ |u-v| \geq c$

Register Separation

Establishing the converse

Claim

If the registers are not k-separable, then $k-1$ registers suffice

Non-Separation: $\forall q, \exists c, \forall \sigma, \exists u, v, |u-v| < c$

Register Separation

Establishing the converse

Claim

If the registers are not k-separable, then $k-1$ registers suffice

Non-Separation: $\exists c, \forall q, \forall \sigma, \exists u, v, |u - v| < c$

Register Separation

Establishing the converse

Claim

If the registers are not k-separable, then $k-1$ registers suffice

Non-Separation: $\exists c, \forall \sigma, \forall q, \exists u, v, |u-v| < c$

Register Separation

Establishing the converse

Claim

If the registers are not k-separable, then $k-1$ registers suffice

Non-Separation: $\exists c, \forall \sigma, \forall q, \exists u, v, \ |u-v|<c$

Register Separation

Establishing the converse

Claim

If the registers are not k-separable, then $k-1$ registers suffice

Non-Separation: $\exists c,\ \forall \sigma,\ \forall q,\ \exists u, v,\ |u - v| < c$

$$\exists c, \forall \sigma, \bigwedge_{q} \bigvee_{u,v} |u - v| < c$$

Register Separation

Establishing the converse

$$\exists c, \forall \sigma, \bigwedge_{q} \bigvee_{u,v} |u - v| < c$$

Register Separation

Establishing the converse

$$\exists c, \forall \sigma, \bigwedge_{q} \bigvee_{u,v} |u - v| < c$$

$(q, \langle u, v \rangle, d_{uv})$

- Says $u - v = d_{uv}$, where $-c < d_{uv} < c$
- Wherever we see "v", replace with "$u - d_{uv}$"

Register Separation

Establishing the converse

$$\exists c, \forall \sigma, \bigwedge_{q} \bigvee_{u,v} |u - v| < c$$

$$(q, \langle u, v \rangle, d_{uv}) \longrightarrow (q', \langle \_, \_ \rangle, \_)$$

- Says $u - v = d_{uv}$, where $-c < d_{uv} < c$
- Wherever we see "v", replace with "$u - d_{uv}$"

Register Separation

Establishing the converse

$$\exists c, \forall \sigma, \bigwedge_{q} \bigvee_{u,v} |u - v| < c$$

$$(q, \langle u, v \rangle, d_{uv}) \longrightarrow (q', \langle \_, \_ \rangle, \_)$$

$$(-c < u' - v' < c) \vee \dots$$

- Says $u - v = d_{uv}$, where $-c < d_{uv} < c$
- Wherever we see "v", replace with "$u - d_{uv}$"

Register Separation

Establishing the converse

$$\exists c, \forall \sigma, \bigwedge_{q} \bigvee_{u,v} |u - v| < c$$

$$(q, \langle u, v \rangle, d_{uv}) \xrightarrow[]{\substack{u' := u'' + 2 \\ v' := v'' + 3}} (q', \langle \_, \_ \rangle, \_)$$

$$(-c < u' - v' < c) \vee \ldots$$

- Says $u - v = d_{uv}$, where $-c < d_{uv} < c$
- Wherever we see "v", replace with "$u - d_{uv}$"

Register Separation

Establishing the converse

$$\exists c, \forall \sigma, \bigwedge_{q} \bigvee_{u,v} |u - v| < c$$

$$(q, \langle u, v\rangle, d_{uv}) \xrightarrow[]{\substack{u' := u'' + 2 \\ v' := v'' + 3}} (q', \langle \_, \_\rangle, \_)$$

$$(-c + 1 < u'' - v'' < c) \vee \ldots \Longleftarrow (-c < u' - v' < c) \vee \ldots$$

- Says $u - v = d_{uv}$, where $-c < d_{uv} < c$
- Wherever we see "v", replace with "$u - d_{uv}$"

Register Separation

Establishing the converse

$$\exists c, \forall \sigma, \bigwedge_{q} \bigvee_{u,v} |u - v| < c$$

$$\left(q, \left\{ \begin{array}{cc} \langle u, v\rangle, & d_{uv} \\ \langle u'', v''\rangle, & d_{u''v''} \end{array} \right\}\right) \xrightarrow{\substack{u' := u'' + 2 \\ v' := v'' + 3}} (q', \langle \_, \_\rangle, \_)$$

$$(-c + 1 < u'' - v'' < c) \vee \ldots \Longleftarrow (-c < u' - v' < c) \vee \ldots$$

- Inductive backpropagation!
- Invariants maintained in DNF form

Register Separation

Establishing the converse

$$\exists c, \forall \sigma, \bigwedge_{q} \bigvee_{u,v} |u - v| < c$$

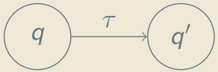
$$\left(q, \left\{ \begin{array}{cc} \langle u, v \rangle, & d_{uv} \\ \langle u'', v'' \rangle, & d_{u''v''} \end{array} \right\}\right) \xrightarrow{\begin{array}{l} u' := u'' + 2 \\ v' := v'' + 3 \end{array}} (q', \langle u', v' \rangle, d_{u''v''} - 1)$$

$$(-c + 1 < u'' - v'' < c) \vee \ldots \Longleftarrow (-c < u' - v' < c) \vee \ldots$$

- Inductive backpropagation!
- Invariants maintained in DNF form

Register Separation

Establishing the converse



Register Separation

Establishing the converse

$$\mathtt{INV}(q) := \mathtt{INV}(q) \wedge \mathtt{WP}(\mathtt{INV}(q'), \tau)$$

$$q \xrightarrow{\tau} q'$$

Repeat at each transition τ until fixpoint

Register Separation

Establishing the converse

$$\mathtt{INV}(q) := \mathtt{INV}(q) \wedge \mathtt{WP}(\mathtt{INV}(q'), \tau)$$

$$q \xrightarrow{\tau} q'$$

Repeat at each transition τ until fixpoint

Claim

A fixpoint will eventually be reached

Register Separation

Final result

Theorem

The register complexity is at least k iff the registers are k-separable

Register Separation

Final result

Theorem

The register complexity is at least k iff the registers are k-separable

Theorem

Computing the register complexity is PSPACE-complete

Conclusion

Conclusion

What we talked about

- Described CRAs as a model for regular functions
- Introduced register separation in CRAs
- Outlined connection between separation and register complexity

Conclusion

What we *didn't* talk about, i.e. what else is in the paper

- Machine-independent characterization of the register complexity
- Analysis of adversarial games over CRAs – optimal reachability
  Undecidable when domain is $\mathbb{Z}$
  EXPTIME-complete when domain is $\mathbb{N}$
- Proofs!

Conclusion

What's left to do

- Understanding register separation in models with binary addition, SSTs, etc.
- Optimal reachability in probabilistic variants
- Variants for ω-strings / trees / ...

Thank you! Questions?

Reserve Slides

Reserve Slides

Gotchas

Gotchas

Bounded registers

- Definition engineering; claims remain true in spirit
- Consider hypothetical “constant-0” register

Gotchas

- Paper assumes $\mathbb{Z}$; also holds for $\mathbb{N}$
- Free algorithm for $\mathbb{Q}$: The rationals admit a notion of "GCD"

Conjecture

- *Similar results hold for $\mathbb{R}$ as well*
- *Can be easily generalized to any commutative group*

Reserve Slides

Weighted automata

Regular Functions / Cost Register Automata

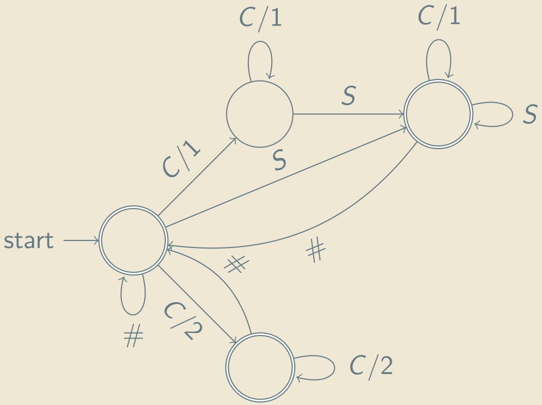
Connection to weighted automata

Use non-determinism!

Regular Functions / Cost Register Automata

Connection to weighted automata

Use non-determinism!



Regular Functions / Cost Register Automata

Connection to weighted automata

- CRAs are equivalent to unambiguous WA
- CRA $(\min, +c)$ equivalent to (full) WA
  $x := \min(x, y), \; y := z + 3$
- Weighted automata are inherently non-deterministic

Fin!