Algebraic Decomposition of Finite State Automata and Formal Models of Understanding

University of Hertfordshire, United Kingdom
Outline

- Krohn-Rhodes Theory
- Formal models of understanding, applications
- Implementations
- Examples
Automata are studied as algebraic objects (semigroup) and the main concern is the 'factorization' of automata.

Partially and informally:

**Theorem 1 (Krohn-Rhodes Decomposition Theorem)**

A finite automaton $\mathcal{A}$ can be represented homomorphically by a cascade product of components from $\{\mathcal{A}_F, \mathcal{A}_{G_1}, \ldots, \mathcal{A}_{G_n}\}$. where $F$ is the flip-flop monoid (the smallest semigroup with an identity and two right-zero element), and $G_1, \ldots, G_n$ denote simple groups dividing the characteristic semigroup of $\mathcal{A}$. 
hierarchical composition, wreath product

\( f_1 \in S_1 \rightarrow (A_1, S_1) \rightarrow b_1 \in A_1 \)

\( f_2 : A_1 \rightarrow S_2 \rightarrow (A_2, S_2) \rightarrow b_2 \in A_2 \)

\( f_3 : A_2 \times A_1 \rightarrow S_3 \rightarrow (A_3, S_3) \rightarrow b_3 \in A_3 \)

\((A_3, S_3) \circ (A_2, S_2) \circ (A_1, S_1) \)

\((a_3, a_2, a_1) \cdot (f_3, f_2, f_1) = (b_3, b_2, b_1) = \)

\((a_3 \cdot f_3(a_2, a_1), a_2 \cdot f_2(a_1), a_1 \cdot f_1) \)
The significance of the theorem

Rephrasing the theorem for a practical computer scientist:

For all systems for which we can give a finite state automata description a hierarchical model can be generated automatically.

Hierarchical implies:

- information flow between levels are restricted enabling modularity (also within one level with parallel components)
- generalization and specialization are natural operations realized by taking subsets of levels in either direction up or down the hierarchy
Practical Applications

- artificial intelligence: creating representations of the environment on the fly
- automated object-oriented programming
- biology: well-defined complexity measure, understanding metabolic networks
- physics: top level coordinates correspond to conserved quantities

The possible users can equally be humans, robots or software.
A mathematical proof sometimes provides a clear algorithm but usually efficiency and computational feasibility are not considered.

Problematic points: \( \exists, \forall \) when the sets are huge. The only problem in semigroup theory: there are so many elements...

A finite state machine with \( n \) states may end up with a semigroup with \( n^n \) elements, e.g. \( 10^{10} = 10 \) billion and sometimes we deal with subsets, and their number is \( 2^{10^{10}} \), so it’s not just the question of memory-upgrade...
Implementations I.

$V \cup T$-technique (Krohn, Rhodes 1965.)

- iteratively decomposes a semigroup into two possibly overlapping parts (using the Green class picture)
- the number of hierarchical components is big $\Rightarrow$ practically inapplicable
- it is implemented in a computer algebra system, called GAP.
Implementations II.


- works with a detailed study of how the automaton’s characteristic semigroup acts on the subsets of the state set
- implemented using breadth-first search, $O(n^n)$
- applying techniques from the theory of formal languages, $O(2^n)$
- implemented as a standalone software (due to its experimental nature) in Java.
Example I

state set: residue classes mod 6
input symbols: adding one +1, doubling x2
$\mathcal{I} = \text{images } \cup \text{ singletons } \cup \text{ state set.}$

The images as they are generated:

$\{0,1,2,3,4,5\}$

$x2$

$\{0,2,4\}$

$+1$

$\{1,3,5\}$
Example I - subduction, tiling, skeleton

\[ P \leq Q \iff \exists s \in S^1, P \subseteq Q \cdot s \quad (P, Q \in \mathcal{I}). \]

\[ P \equiv Q \iff P \leq Q \text{ and } Q \leq P. \]

\[ P < Q, \text{ if } P \subset Q, P < Q \text{ and } P \text{ is maximal} \]

\[ C_2 \wr S_3 \]
Example II – randomly generated automaton
The bottleneck of the current implementation is that $\mathcal{I}$ is fully calculated.
Example II – long decomposition

15 hierarchical levels for 6 states
Thank You!

More information available at:

http://graspermachine.sf.net

REMEMBER!!!
If You have finite state automata then we can tell You how to understand them exactly! *

* For the time being up to 10 states for an automaton.