Automata Theory for Presburger Arithmetic Logic

References from
Introduction to Automata Theory, Languages & Computation
and
Constraints in Computational Logic Theory & Application
Presented by Masood Obaid 18th Jan. 2002
International Max Planck Research School Saarbrucken.
Structure of Presentation

- Introduction of notions and automata theory
  - Formal definitions and examples
- Presburger Arithmetic
- Translation from Presburger formulas to finite automata
  - Automata associated with equalities and inequalities
- Extensions....
Introduction to notions

- Strings
  - Prefix
  - Suffix
  - Concatenation
    - Empty string
Alphabets
- An Alphabet is a finite set of symbols

Language
- \( \Sigma = \{a\} \) then \( \Sigma^* = \{e, a, aa, aaa, \ldots, \ldots, \ldots\} \)
Graph

- Mathematically written as $G (V, E)$
- Path
  - $v_1, v_2, \ldots, v_k, k \geq 1$, such that there is an edge $(v_i, v_{i+1})$ for each $i, 1 \leq i < k$
- Cycle
  - if $v_i = v_k$ the path is a cycle
Directed Graph

- Also called as digraph is a sequence of vertices and an ordered pair of edges called arcs.
- \( v_i \rightarrow v(i+1) \) is an arc for each \( i, 1 \leq i < k \).
- An arc from \( x \) to \( y \) is denoted by \( x \rightarrow y \). In the arc \( x \rightarrow y \), \( x \) is the predecessor of \( y \) and \( y \) is the successor of \( x \).
Tree

- There is one vertex called the root that has no predecessors and from which there is a path to every vertex
- Each vertex other than the root has exactly one predecessor
- The successors of each vertex are ordered from the left
Inductive Proofs

- Principle of Mathematical induction is as follows
  - Basic Step \( P(0) \)
  - Inductive Step \( P(n) \) implies \( P(n+1) \) for \( n \geq 0 \)

- Example
  - To prove that \( 0+1+2+3+\ldots+n=n(n+1)/2 \)
Proof

• We could argue like this: For \( n = 0 \), the result is clearly true i.e., \( 0 = 0.2 / 2 = 0 \)

Now suppose that for integer \( n \geq 1 \),
\[
0 + 1 + 2 + \ldots + n = \frac{n(n+1)}{2}
\]

By induction hypothesis
\[
(1 + 2 + \ldots + n) + (n+1) = \frac{n(n+1)}{2} + (n+1)
\]
\[
= \frac{n(n+1)+2(n+1)}{2}
\]
\[
= \frac{(n+1)(n+2)}{2} = \frac{(n+1)(n+1+1)}{2}
\]

and so the result follows by induction.
Sets

- A set is a collection of objects without repetition.

Set notation

- \{x \in A \mid p(x)\} “The set of objects x such that p(x) is true”.
Operations On Sets

- Union ‘∪’ \( \{x| x \in A \text{ or } x \in B\} \)
- Intersection ‘∩’ \( \{x| x \in A \text{ and } x \in B\} \)
- Difference ‘—’ \( \{x| x \in A \text{ and } x \not\in B\} \)
- Product ‘ X’ \( \{(a,b) | a \in A \text{ and } b \in B\} \)
- Power Set is the set of all subsets of A. Denoted by \(2^A\)
Relation

- Domain
- Range

Properties of Relations

- Reflexive: if \(aRa\) for all \(a\) in set \(\{(a,a),(b,b)\}\)
- Irreflexive: if \(aRa\) is false for all \(a\) in set \(\{(a,b)\}\)
- Transitive: if \(aRb\) and \(bRc\) imply \(a Rc\) \(\{(a,b),(b,c),(a,c)\}\)
- Symmetric: if \(aRb\) implies \(bRa\) \(\{(a,b),(b,a)\}\)
- Asymmetric: if \(aRb\) implies \(bRa\) is false \(\{(a,b),(b,c)\}\)
Finite State System

- Linear Control Systems
  - Control Mechanism of an elevator
- Telecommunication
  - Pay phone controller
- Computer Science
  - Compilers
  - Switching circuit (control unit of a computer)
Finite Automaton

- Formally a finite automaton is defined by a 5-tuple \((Q, A, \delta, q_0, Q_f)\)
  - \(Q\) is the finite set of states.
  - \(A\) is the input alphabet.
  - \(\delta\) is the transition relation mapping \(Q \times A\) to \(Q\),
    - i.e., \(\delta(q, a)\) is a state for each state \(q\) and input symbol \(a\).
  - \(q_0 \in Q\) is the initial state,
  - \(Q_f \subseteq Q\) is the set of Final states.
Transition Diagram

- A directed graph that represents finite automaton is called Transition Diagram.
- States of the automaton
- Transition from one state to another
In this transition diagram of finite automaton, the control is at $q_0$ if and only if there are both an even number of 0’s and even number of 1’s in the input sequence.
$M = (Q, A, \delta, q_0, Q_f)$.  

- Where $Q = \{q_0, q_1, q_2, q_3\}$ and $A = \{0, 1\}$, $Q_f = \{q_0\}$ and $\delta$ is shown in the following table.

<table>
<thead>
<tr>
<th>States</th>
<th>Inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>0</td>
</tr>
<tr>
<td>$q_1$</td>
<td>$q_2$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_0$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_1$</td>
</tr>
</tbody>
</table>

Applying 110101 as input sequence to $M$. We get $q_0$. 
Finite Automaton Continued…

- For every word $w \in A^*$ if $i$ is a natural number which is smaller or equal to the length of $w$ we can say $w(i)$ is the $i^{th}$ letter of word $w$.

- A run of the automaton $(Q, Q_f, q_0, \delta)$ on a word $w \in A^*$ is a state sequence $\rho \in Q^*$ such that $\rho(0) = q_0$, and if $\rho(i) = q$, $\rho(i+1) = q'$ then $(q, w(i), q') \in \delta$.

- **Successful Run**
  
  - A successful run $\rho$ on $w$ is a run such that $\rho(n+1) \in Q_f$ where $n$ is length of $w$. 
Non Deterministic Finite Automata

- Definition
- An Example
  I/P sequence 01001 is accepted by the NFA. The sequence of transitions is through the states $q_0, q_0, q_0, q_3, q_4, q_4$
- Difference
Finite Automata with output

- **Moore Machine**

Represented by tuple \((Q, A, \delta, q_0, Q_f, \lambda, \Delta,)\) where \(Q, A, \delta, q_0, Q_f\) have their usual meanings as in FA. \(\Delta\) is the output alphabet and \(\lambda\) is mapping from \(Q\) to \(\Delta\) giving the output associated with each state.

On input 1010 number of states entered is \(q_0, q_1, q_2, q_2, q_1\) giving output sequence 01221.

i.e., 0 has residue 0, 1 has 1, \(2_{10}\) has 2, \(5_{10}\) has 2 and \(10_{10}\) has 1.
### Mealy Machine

Represented by \((Q, A, \delta, q_0, Q_f, \Delta, \lambda)\) where \(Q, A, \delta, q_0, Q_f\) have their usual meanings as in Moore machine. Except that \(\lambda\) is mapping from \(Q \times A\) to \(\Delta\). i.e., \(\lambda(q,a)\) gives the output associated with the transition from state \(q\) on input \(a\).

Response of \(M\) to input \(01100\) is \(n, n, y, n, y\) with the sequence of states being entered \(q_0, p_0, p_1, p_1, p_0, p_0,\)

Machine \(M(\{q_0,p_0,p_1\}, \{0,1\}, \delta, \{q_0\}, \{y,n\}, \lambda)\)
Presburger Arithmetic

- **Basic terms**
  - \(x+x+1+1+1\) is an example which is a basic term and can be abbreviated as \(2x+3\).

- **Atomic formula**
  - are equalities and inequalities between basic terms. For instance \(x+2y=3z+1\) is an atomic formula.

- **Formulas**
  - of the logic are the first-order formula built on the atomic formulas.
- **Connectives**
  - Conjunction (\(\land\))
  - Disjunction (\(\lor\))
  - Negation (\(\neg\))
  - Existential Quantification (\(\exists\))
  - Universal Quantification (\(\forall\))

- An example of a logical formula is \(\forall x (\exists y x=2y \lor \exists y x=2y+1)\) Expressing in words as, every integer is even or odd.
**Free Variables**

- Variables which are not quantified are called free variables

**Solution**

- A solution of a formula $\varphi(x_1, x_2, \ldots, x_n)$ is an assignment of $x_1, x_2, \ldots, x_n$ in $\mathbb{N}$ which satisfies the formula. For instance $\{x=0; y=2; z=1\}$ is a solution of $x+2y=3z+1$ and every assignment $\{x=n\}$ is a solution of, $\exists y (x=2y \lor x=2y+1)$
Translation from Presburger
Formula to Finite Automata

- Automata associated with equalities

For every basic formula

\[ a_1x_1 + a_2x_2 + \ldots + a_n x_n = b \] (where \( a_1, a_2, \ldots, a_n, b \in \mathbb{Z} \))

The automaton is constructed by saturating the set of transition and the set of states, initially set to \( \{ q_b \} \) using the inference rule:

- If \( \theta \in \{0,1\}^n \) encodes \((\theta_1, \ldots, \theta_n)\)

\[
q_c \in Q, \quad a_1 \theta_1 + \ldots + a_n \theta_n = 2c
\]

For every state \( q_c \in Q \), computing the solutions \((\theta_1, \ldots, \theta_n)\) of

\[
a_1x_1 + a_2x_2 + \ldots + a_n x_n = c \mod 2
\]

and add the state \( q_d \) and the rule \( q_c \theta \rightarrow q_d \) where \( d = (c - a_1 \theta_1 - \ldots - a_n \theta_n) / 2 \).
Considering the equation \( x+2y = 3z+1 \)

Here \( b=1 \) we have \( q_1 \in Q \). Computing the solution modulo 2 of the above equation we get \{ (0,0,1), (0,1,1), (1,0,0), (1,1,0) \}.

Computing the new states by

- \( d= (1-0-0+3)/2=2 \) i.e., \( q_2 \)
- \( d= (1-0-2+3)/2=1 \) i.e., \( q_1 \)
- \( d= (1-1-0-0)/2= 0 \) i.e., \( q_0 \)
- \( d= (1-1-2-0)/2= -1 \) i.e., \( q_{-1} \)

- \( q_2, q_1, q_0, q_{-1} \) are \( \in \) of \( Q \).

The new transitions

\[ q_1(0,0,1) \rightarrow q_2 \quad q_1(0,1,1) \rightarrow q_1 \quad q_1(1,0,0) \rightarrow q_0 \quad q_1(1,1,0) \rightarrow q_{-1} \]
Automaton for equation $x+2y = 3z+1$
In this automaton our initial state is $q_1$ and final state is $q_0$ indicated by double circle. Traversing the transition diagram from initial state $q_1$ through $q_{-1}$, $q_{-2}$, $q_0$, $q_1$, $q_2$ to the final state $q_0$ we get the word

\[
\begin{array}{cccccccc}
# & q_1 & \rightarrow q_{-1} & \rightarrow q_{-2} & \rightarrow q_0 & q_0 & \rightarrow q_1 & \rightarrow q_2 & \rightarrow q_0 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\
2 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\
3 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
\end{array}
\]

#1,2 & 3 gives $x=15_d$, $y=35_d$ & $z=28_d$

The word is accepted by the automaton and verifies the equation $x+2y=3z+1 \implies 15+2(35)=3(28)+1 \implies 85=85$
Automata associated with inequalities

- Computing an automaton for inequality

\[ a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq b \]

Starting from state \( q_b \) as initial state and from a state \( q_c \) we compute the transitions and states as:

\[ q_c \xrightarrow{\theta_1 \ldots \theta_n} q_d \]

with \( d = \left\lfloor (c - \sum_{i=1}^{n} a_i \theta_i)/2 \right\rfloor \)

\[ Q_f = \{ q_c \mid c \geq 0 \} \]
- Considering the inequality $2x-y \leq -1$ we get the following automaton

```
                      (-2)
 (0,1)          (0,0)

                      (-1)
 (1,0)          (1,1)

                      (0)
 (0,1)          (1,0)

```

Traversing the machine from initial state $q_{-1}$ through $q_{-1}, q_{-2}, q_{-2}, q_{-1}, q_{0}$ to the final state $q_{0}$ we get the word

```
# $q_{-1} \rightarrow q_{-1} \quad q_{-1} \rightarrow q_{-2} \quad q_{-2} \rightarrow q_{-1} \quad q_{-1} \rightarrow q_{0} \quad q_{0} \rightarrow q_{0}$
1 1 1 0 0 0 0
2 1 0 1 1 1
```

#1 & 2 gives $x=3_{d}$ & $y=29_{d}$

The word is accepted by the automaton and verifies the equation $2x-y \leq -1 \Rightarrow 2(3) - 29 = 6 - 29 = -23 \leq -1$
Closure properties

- A class of languages closed under a particular operation
- Union and intersection i.e., $A_{\varphi_1} \lor A_{\varphi_2}$, $A_{\varphi_1} \land A_{\varphi_2}$ of two automata $A_{\varphi_1}(\vec{x})$ and $A_{\varphi_2}(\vec{x})$ can be computed by the classical constructions.
- Negation corresponds to complement. For quantifier $\forall x \varphi$ the complement is $\neg \exists x \neg \varphi$. We only have to consider existential quantification. $A_{\exists x \varphi}$ is computed from $A_{\varphi}$ by projection. Transitions, Initial and Final states are identical to those of $A_{\varphi}$. 
Example

Automaton accepting solution for

$$\exists z \; x+2y=3z+1$$
Extensions...

- More expressive automata models are
  - Push down automata (Stacks)
  - Turing Machines
  - Rabin Automata