Automata theory

An algorithmic approach

Automata as data structures

- Data structures allow us to represent sets of objects in a computer.
- Different data structures support different sets of operations (dictionary, stack, queue, priority queue, ...):

Op. set	Operations	Data structures
Dictionary	insert, lookup, remove	Hash tables, arrays, search trees
Stack	push, pop	Linked list, array
Priority queue	insert_with_priority, extract_highest_priority	Heap, binomial heap, Fibonacci heap
Union-find	set union, find set	Linked lists, disjoint forests

Automata as data structures

- In this course we look at automata as a data structure supporting
  - the boolean operations of set theory (union, intersection, complement with respect to a given universe set)
  - property checks (emptiness, universality, inclusion, equality)
  - operations on relations (projections, joins, pre, post)

In more detail

Member(x, X)	:	returns **true** if $x \in X$, **false** otherwise.
Complement(X)	:	returns $U \setminus X$.
Intersection(X, Y)	:	returns $X \cap Y$.
Union(X, Y)	:	returns $X \cup Y$.
Empty(X)	:	returns **true** if $X = \emptyset$, **false** otherwise.
Universal(X)	:	returns **true** if $X = U$, **false** otherwise.
Included(X, Y)	:	returns **true** if $X \subseteq Y$, **false** otherwise.
Equal(X, Y)	:	returns **true** if $X = Y$, **false** otherwise.
Projection_1(R)	:	returns the set $\pi_1(R) = \{x \mid \exists y\ (x, y) \in R\}$.
Projection_2(R)	:	returns the set $\pi_2(R) = \{y \mid \exists x\ (x, y) \in R\}$.
Join(R, S)	:	returns $R \circ S = \{(x, z) \mid \exists y \in X\ (x, y) \in R \wedge (y, z) \in S\}$
Post(X, R)	:	returns $post_R(X) = \{y \in U \mid \exists x \in X\ (x, y) \in R\}$.
Pre(X, R)	:	returns $pre_R(X) = \{y \in U \mid \exists x \in X\ (y, x) \in R\}$.

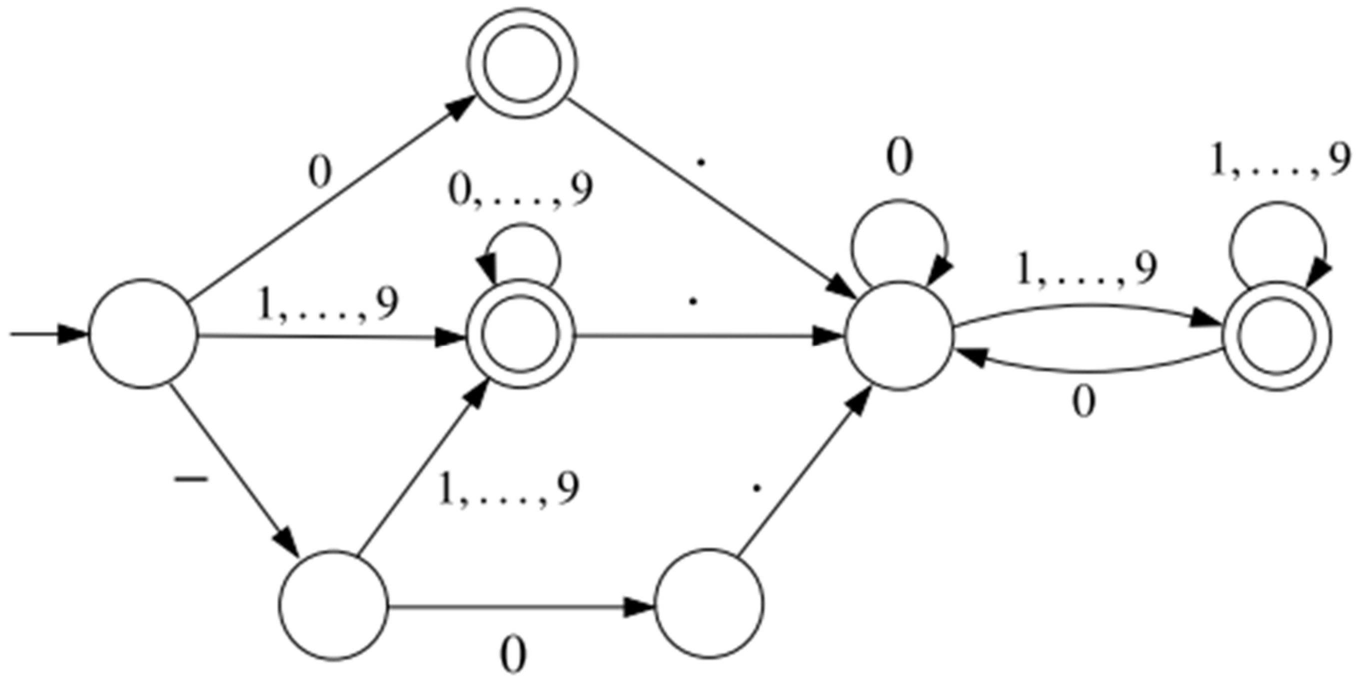
- U denotes some universe of objects (numbers, names, records, ...)
- X, Y denote subsets of U, x denotes an element of U
- R, S denote binary relations on U, i.e., $R, S \subseteq U \times U$

Basic idea

- Elements of the universe can be encoded as words (strings over some alphabet)
- Sets can be encoded as languages (sets of words)
- Automata recognize languages.

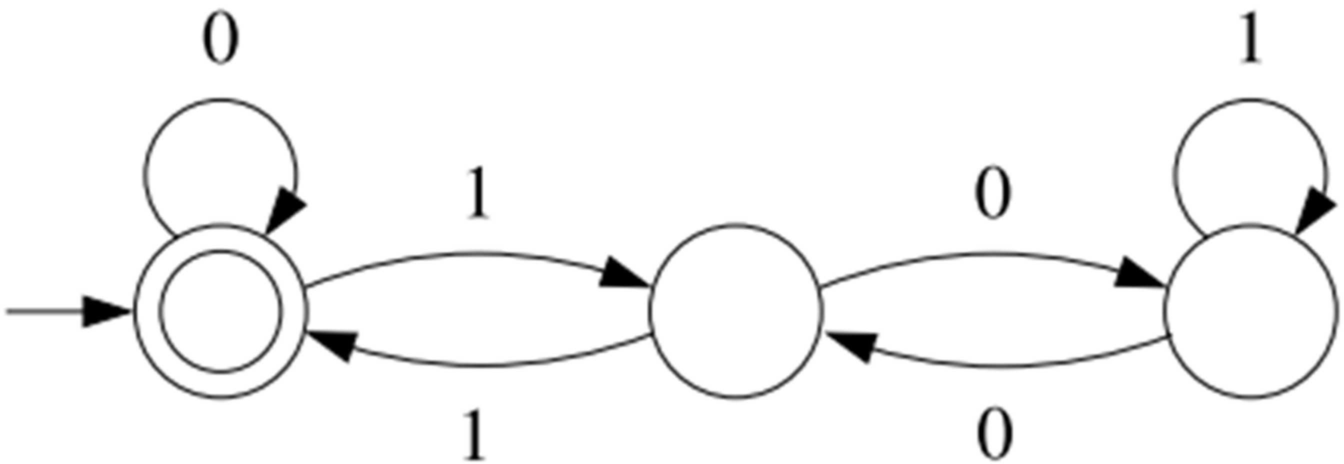
Examples

- An automaton for the strings encoding decimal numbers



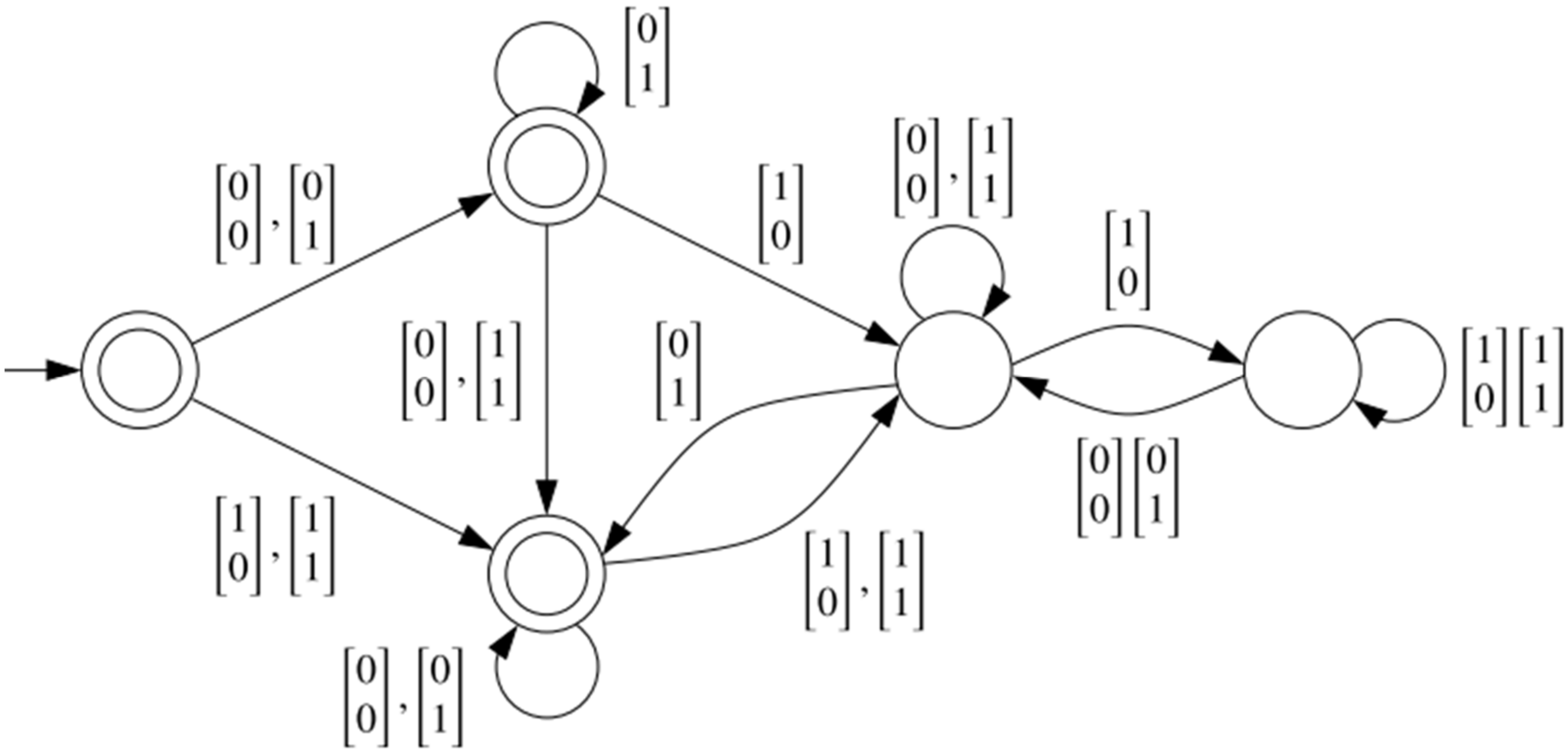
Examples

- An automaton for the multiples of 3 in binary.



Examples

- An automaton for the nonnegative solutions of $2x - y \leq 2$ in binary (least significant bit first)



Examples

- An automaton for the reachable configurations of a program

```
while x = 1 do
   if y = 1 then
      x ← 0
   y ← 1 − x
end
```