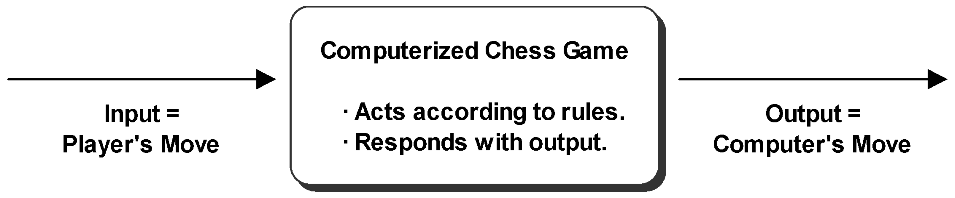
1. Game theory is represented in the theory of automata

Take the example of a computerized chess game a human player inputs their moves into the computerized system. Depending on the state of the system, it responds with an output move based on rules of operation, which include strategies of chess and the algorithm that each move towards winning equals objectives achieved. At the beginning of the game, the system is at an initial state. As the game progresses, each possible move on the board leads to a new state in the computer. The computerized chess game is an automaton.



2. What is an automaton?

An automaton may be considered, abstractly, as a set of physically unspecified states, inputs, outputs, and rules of operation, and the study of automata is the investigation of what can be accomplished with these states. Automata are sequential machines and are, therefore, self-operational; automata function without human participation.

A sequential machine consists of two main structures (Bavel, 1983):

1) The **transition structure** includes the states, the inputs, and the parts necessary to determine the transitions of the machine, including rules of operation, or algorithms (if this state and this input, then this output).
2) The **output structure** includes mainly the states, the outputs, and the output function. The output structure is dependent on the transition structure and is determined by the designer.

3. Algorithms and Automata

Algorithm

An algorithm is a procedural set of instructions that is comprehensive. Automata execute algorithms.

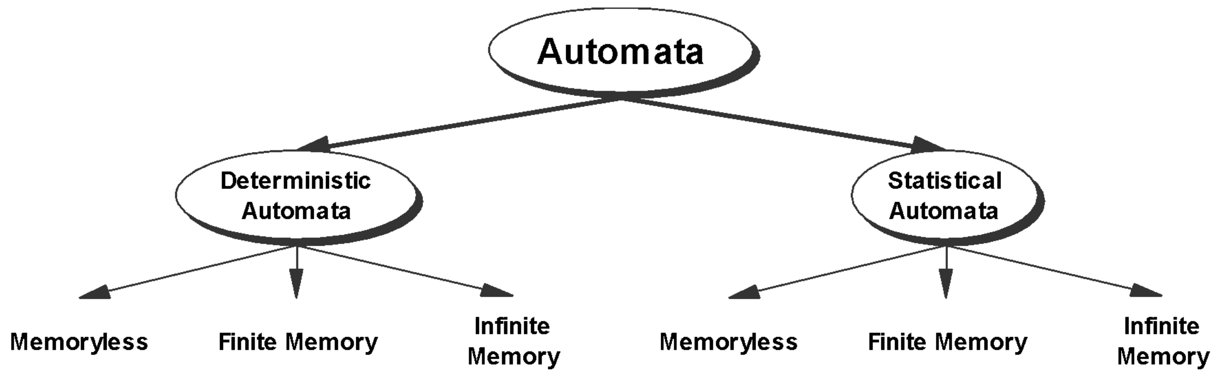
Algorithms determine the content and the sequence of operations for transforming the initial data into the sought result.

Any algorithm should satisfy the following requirements (Lerner, 1972):

- *Definiteness:* accuracy and uniqueness without leaving room for arbitrariness.
- *Generality:* the algorithm can be used for an entire class of problems not just for the same problem with different inputs.
- *Effectiveness:* an algorithm leads to the sought results after performing a finite number of operations.

4. Categories and Types of Automata

There are two categories of automata and three types of automata that are present in each of these categories.



Categories

1) Deterministic
Output is uniquely determined by the input sequences. Given any input, a definite output results.

The behaviour of such automata can be accurately predicted if the transfer operator is known and given in the form of a table of a logical function, and if the initial state and the input sequence are also known.

Examples: teaching machine, combination lock, point of sale scanner, computerized chess game

2) Statistical (probabilistic or stochastic)
Random output sequences are generated given any fixed input. The amount of randomness can be set by applying a probability of any output given the system's current state and input sequence.

Example: student, slot machine

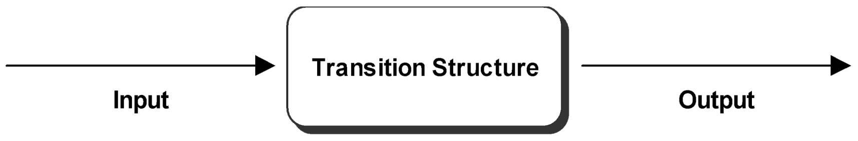
Types

There are three basic **types** of automata, as classified by the amount of input they process before yielding an output:

1) Memoryless

This type of automata recognizes only one input at a time and produces output based on that input. The output is not influenced by any additional inputs that arrived beforehand. The reaction time of the automaton is constant for all input signals. The internal state of such an automaton is independent from any external action.

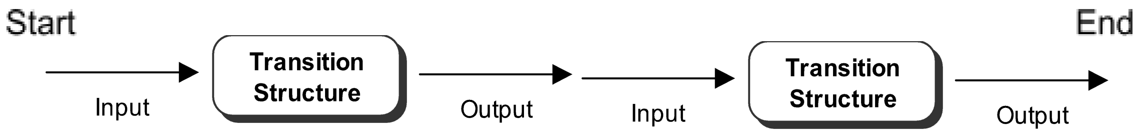
Example: point of sale scanner, slot machine



2) Finite Memory

The group of output signals generated at a given time depends not only on the signals applied at the same moment, but also on those that arrived earlier. These preceding external actions (or fragments of them) are recorded in the automaton by a variation of its internal state. The reaction of such an automaton is uniquely determined by the group of input signals that has arrived and by its internal state at a given time. These factors also determine the state into which the automaton goes.

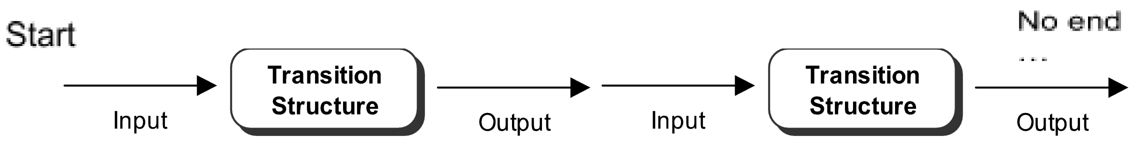
Example: teaching machine, combination lock, computerized chess game



3) Infinite Memory

Infinite memory comprises an **abstract** circuit of a logical automaton, which in principle is suitable for realizing any information processing algorithm. Infinite automata can process 'more' than finite automata because infinite automata can continue to move from state to state without having to revert to a previous state, since there are an infinite number of states. When only a finite number of states exists, it becomes necessary to return to states already traversed to keep moving. The Turing machine belongs to this type of automata.

Example: student, computer, the Turing machine



5. Turing Machine: The Beginning of Infinite Memory Automata

What is a Turing Machine?

In 1936–7, Alan Turing, an English mathematician, conceptualized a machine with **infinite memory** that could **realize any algorithm**. He believed that, if a set of rules describing a computation could be written down, then his machine could execute those rules. Turing's Machine is the cornerstone of the modern theory of computation and computability even though it was invented nine years before the creation of the first electronic digital computer.

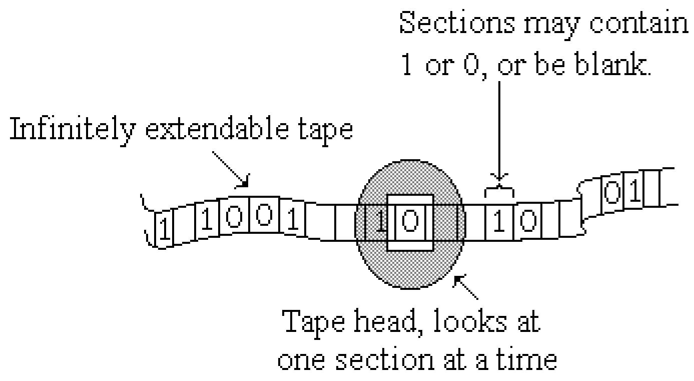
The Turing Machine consists of…

- an Input/Output Tape,
- the Turing Machine itself,
- and a Rule List.

The **Input/Output Tape** is like the roll of paper found on some printing calculators; only this roll of paper is infinitely long and is stretched like a scroll between two rollers so it can be wound forward and backward. The tape is divided into cells. The cells contain the input and output symbols and change frequently as a program is running.

The **Rule List** is what determines the Machine's move at any particular point.

How does it work?


Sections may contain 1 or 0, or be blank.
Infinitely extendable tape
Tape head, looks at one section at a time

The Turing Machine reads a symbol from the Input/Output Tape and consults its Rule List. It then performs two actions.

1. It modifies its internal State.

2. It writes a symbol on the tape only on the section being viewed.

or

2. It moves its Read-Write head left or right.

The machine reads its input from the tape, refers to rules controlling its behaviour, and considering both the input and its own current state, determines what behaviour to exhibit (i.e. erase/write on tape, move tape) and which internal state to assume next.

Example:
Suppose the machine is in State 10 and the Read/Write head is positioned on '0'. The Turing Machine would consult its Rule List and possibly find the following rule:

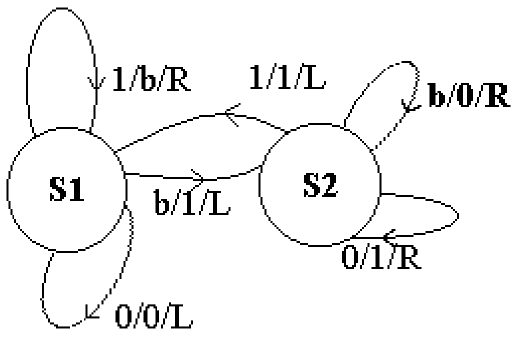
10 '0' 12 >

This rule says: If you are in State 10 and reading '0', change to State 12 and advance the tape head to the right.

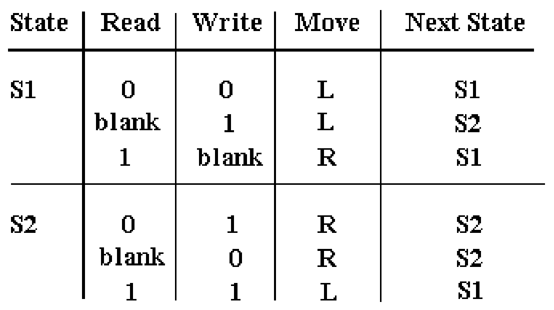
The operation of the Turing machine is uniquely determined by:

a) the initial filling of the cells of the tape and
b) the transformation operator of the control automaton, which can be given in the form of a table of transitions.

Any particular Turing Machine can be described as a chart, describing the actions set for each state, or as a State Transition Diagram, representing the same information in diagrammatic form.



State Diagram of Turing Machine



State	Read	Write	Move	Next State
S1	0	0	L	S1
	blank	1	L	S2
	1	blank	R	S1
S2	0	1	R	S2
	blank	0	R	S2
	1	1	L	S1

State Transition Table for a Turing Machine

If the table of transitions is appropriately expanded, the Turing machine can be used for solving problems of any complexity, and also for a problem that can be solved by any other machine; hence, it is called Universal Turing Machine. At the time of its conception, the use of automata built according to the Turing principle was found impractical because the number of steps required for solving very complicated problems was extremely large, which meant that a very long solution time was required. With the speed of current digital computers, this is no longer a problem for many algorithms.

The theory of Turing machines is of great importance for describing what functions can and what functions cannot be performed automatically, in particular by such universal automaton as the digital computer. It contributes to the beginning of the modern theory of the digital computer.

6. The Teaching Machine: An Example of Deterministic Finite Automata

Bung and Lansky (1978) presented an example from Fierli (1974) of an automaton modelling the behaviour of a teacher. We imitate this example by presenting an automaton as a teaching machine that executes the teaching algorithms presented by Bung and Lansky.

This teaching machine confines its activities to presenting the student with tasks taken from two sets:

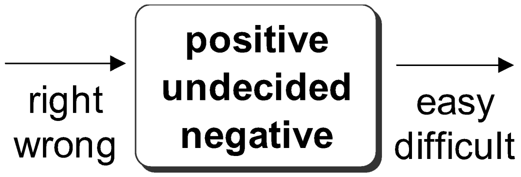
set d, containing *difficult* tasks; and
set e, containing *easy* tasks.

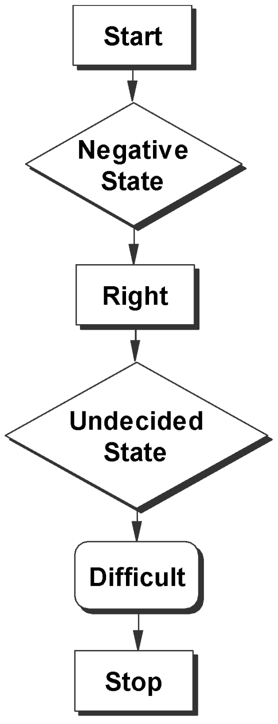
The corresponding automaton A, therefore, has an output alphabet Z = (d, e). The teaching machine receives the student's responses and marks them

either *right* (r) or
wrong (w)

A, therefore, has an input alphabet X = (r, w). The teaching machine tries to decide on whether the student has reached his objectives or not. In this respect, the teaching machine can be in one of three states,

positive (p)
undecided (u) or
negative (n)





That is A has a state alphabet Z = (p, u, n). The teaching machine behaves in accordance with the following teaching algorithm: when it is in state n and receives a wrong response, it presents the next task from set e and remains in state n. When it is in state n and receives a correct response, it moves to state u (one step towards *positive* = objectives achieved), but still presents the next task from set e. When it is in state u and receives a wrong response, it moves to state n and presents a task from set e. When it is in state u and receives a correct response, it moves to state p and presents a task from set d. When it is in state p and receives a wrong response, it moves to state u and presents a task from set d. When it is in state p and receives a correct response, it decides that the student has achieved his objectives and stops presenting further tasks (formally, the automaton puts out the empty letter Φ and remains in state p). This teaching algorithm can be unambiguously described through the automaton table:

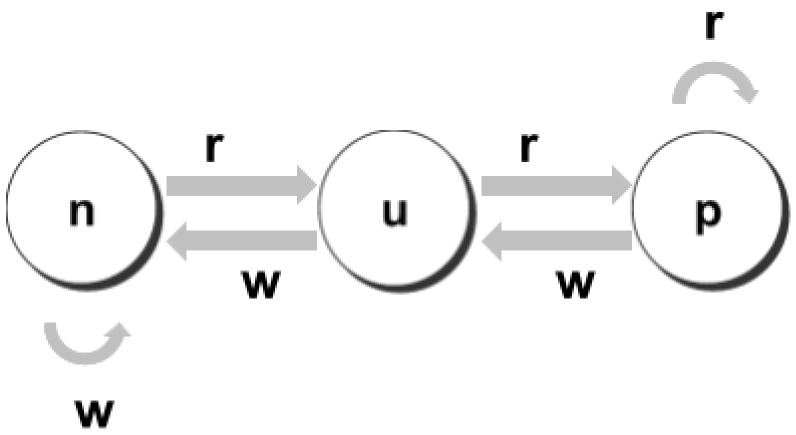
Transition Matrix and Output Matrix Combined

s t a t e z

input x	**p**ositive	**u**ndecided	**n**egative	
right	**p / Φ** objective achieved	**p/d**	**u / e**	
wrong	**u/d**	**n/e**	**n/e**	
				δ/ λ

δ - new state
λ - output

State Diagram of Teaching Machine



$$\begin{tikzcd} n \arrow[loop below, "w"] \arrow[r, shift left, "r"] & u \arrow[l, shift left, "w"] \arrow[r, shift left, "r"] & p \arrow[l, shift left, "w"] \arrow[loop above, "r"] \end{tikzcd}$$

7. Purpose

How is the theory of automata useful for educational technologists?

- Is useful for designing flowcharts for hypermedia and simulations.
- Explores how machines work.

How does the theory of automata fit into cybernetics?

- The theory of automata is a branch of communication, which explains its relation to cybernetics.

Sources on Automata

Bavel, Z. (1983). *Introduction to the theory of automata*. Reston, VA: Reston Publishing Company.
Presents some comprehensible descriptions of automata. Associates state diagrams of automata and transition matrices.

Bung, K., & Lansky, M. (1978). Educational cybernetics. *The encyclopaedia of educational media communications and technology, pp. 266-299. London: MacMillian.*
Discusses the theory of automata in relation to an educational context.

Lerner, A. Y. (1972). *Fundamentals of cybernetics*. London: Chapman and Hall.
Contains chapters on automatic control, automata, and the computer.