8 Advanced Counting techniques

Key Terms and Results

TERMS

recurrence relation: a formula expressing terms of a sequence, except for some initial terms, as a function of one or more previous terms of the sequence

initial conditions for a recurrence relation: the values of the terms of a sequence satisfying the recurrence relation before this relation takes effect

dynamic programming: an algorithmic paradigm that finds the solution to an optimization problem by recursively breaking down the problem into overlapping subproblems and combining their solutions with the help of a recurrence relation

linear homogeneous recurrence relation with constant coefficients: a recurrence relation that expresses the terms of a sequence, except initial terms, as a linear combination of previous terms

characteristic roots of a linear homogeneous recurrence relation with constant coefficients: the roots of the polynomial associated with a linear homogeneous recurrence relation with constant coefficients

linear nonhomogeneous recurrence relation with constant coefficients: a recurrence relation that expresses the terms of a sequence, except for initial terms, as a linear combination of previous terms plus a function that is not identically zero that depends only on the index

divide-and-conquer algorithm: an algorithm that solves a problem recursively by splitting it into a fixed number of smaller non-overlapping subproblems of the same type

generating function of a sequence: the formal series that has the nth term of the sequence as the coefficient of x^n

sieve of Eratosthenes: a procedure for finding the primes less than a specified positive integer

derangement: a permutation of objects such that no object is in its original place

RESULTS

the formula for the number of elements in the union of two finite sets:

$$|A \cup B| = |A| + |B| - |A \cap B|$$

the formula for the number of elements in the union of three finite sets:

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

the principle of inclusion–exclusion:

$$|A_1 \cup A_2 \cup \cdots \cup A_n| = \sum_{1 \le i \le n} |A_i| - \sum_{1 \le i < j \le n} |A_i \cap A_j| + \sum_{1 \le i < j < k \le n} |A_i \cap A_j \cap A_k| - \cdots + (-1)^{n+1} |A_1 \cap A_2 \cap \cdots \cap A_n|$$

the number of onto functions from a set with m elements to a set with n elements:

$$n^m - C(n,1)(n-1)^m + C(n,2)(n-2)^m - \cdots + (-1)^{n-1} C(n,n-1) \cdot 1^m$$

the number of derangements of n objects:

$$D_n = n! \left[1 - \frac{1}{1!} + \frac{1}{2!} - \cdots + (-1)^n \frac{1}{n!} \right]$$

9. Relations

Key Terms and Results

TERMS

binary relation from A to B: a subset of $A \times B$

relation on A: a binary relation from A to itself (i.e., a subset of $A \times A$)

$S \circ R$: composite of R and S

R^{-1}: inverse relation of R

R^n: nth power of R

reflexive: a relation R on A is reflexive if $(a, a) \in R$ for all $a \in A$

symmetric: a relation R on A is symmetric if $(b, a) \in R$ whenever $(a, b) \in R$

antisymmetric: a relation R on A is antisymmetric if $a = b$ whenever $(a, b) \in R$ and $(b, a) \in R$

transitive: a relation R on A is transitive if $(a, b) \in R$ and $(b, c) \in R$ implies that $(a, c) \in R$

n-ary relation on $A_1, A_2, \ldots, A_n$: a subset of $A_1 \times A_2 \times \cdots \times A_n$

relational data model: a model for representing databases using n-ary relations

primary key: a domain of an n-ary relation such that an n-tuple is uniquely determined by its value for this domain

composite key: the Cartesian product of domains of an n-ary relation such that an n-tuple is uniquely determined by its values in these domains

selection operator: a function that selects the n-tuples in an n-ary relation that satisfy a specified condition

projection: a function that produces relations of smaller degree from an n-ary relation by deleting fields

join: a function that combines n-ary relations that agree on certain fields

directed graph or digraph: a set of elements called vertices and ordered pairs of these elements, called edges

loop: an edge of the form (a, a)

lexicographic order: a partial ordering of Cartesian products or strings

Hasse diagram: a graphical representation of a poset where loops and all edges resulting from the transitive property are not shown, and the direction of the edges is indicated by the position of the vertices

maximal element: an element of a poset that is not less than any other element of the poset

minimal element: an element of a poset that is not greater than any other element of the poset

greatest element: an element of a poset greater than all other elements in this set

least element: an element of a poset less than all other elements in this set

upper bound of a set: an element in a poset greater than all other elements in the set

lower bound of a set: an element in a poset less than all other elements in the set

least upper bound of a set: an upper bound of the set that is less than all other upper bounds

greatest lower bound of a set: a lower bound of the set that is greater than all other lower bounds

lattice: a partially ordered set in which every two elements have a greatest lower bound and a least upper bound

closure of a relation R with respect to a property P: the relation S (if it exists) that contains R, has property **P**, and is contained within any relation that contains R and has property **P**

path in a digraph: a sequence of edges $(a, x_1), (x_1, x_2), \ldots, (x_{n-2}, x_{n-1}), (x_{n-1}, b)$ such that the terminal vertex of each edge is the initial vertex of the succeeding edge in the sequence

circuit (or cycle) in a digraph: a path that begins and ends at the same vertex

R^* (connectivity relation): the relation consisting of those ordered pairs (a, b) such that there is a path from a to b

equivalence relation: a reflexive, symmetric, and transitive relation

equivalent: if R is an equivalence relation, a is equivalent to b if $a\,R\,b$

$[a]_R$ (equivalence class of a with respect to R): the set of all elements of A that are equivalent to a

$[a]_m$ (congruence class modulo m): the set of integers congruent to a modulo m

partition of a set S: a collection of pairwise disjoint nonempty subsets that have S as their union

partial ordering: a relation that is reflexive, antisymmetric, and transitive

poset (S, R): a set S and a partial ordering R on this set

comparable: the elements a and b in the poset $(A, \preccurlyeq)$ are comparable if $a \preccurlyeq b$ or $b \preccurlyeq a$

incomparable: elements in a poset that are not comparable

total (or linear) ordering: a partial ordering for which every pair of elements are comparable

totally (or linearly) ordered set: a poset with a total (or linear) ordering

well-ordered set: a poset $(S, \preccurlyeq)$, where $\preccurlyeq$ is a total order and every nonempty subset of S has a least element

compatible total ordering for a partial ordering: a total ordering that contains the given partial ordering

topological sort: the construction of a total ordering compatible with a given partial ordering

RESULTS

The reflexive closure of a relation R on the set A equals $R \cup \Delta$, where $\Delta = \{(a, a) \mid a \in A\}$.

The symmetric closure of a relation R on the set A equals $R \cup R^{-1}$, where $R^{-1} = \{(b, a) \mid (a, b) \in R\}$.

The transitive closure of a relation equals the connectivity relation formed from this relation.

Warshall's algorithm for finding the transitive closure of a relation

Let R be an equivalence relation. Then the following three statements are equivalent: (1) $a\,R\,b$; (2) $[a]_R \cap [b]_R \neq \emptyset$; (3) $[a]_R = [b]_R$.

The equivalence classes of an equivalence relation on a set A form a partition of A. Conversely, an equivalence relation can be constructed from any partition so that the equivalence classes are the subsets in the partition.

The principle of well-ordered induction

The topological sorting algorithm

10. Graphs

Key Terms and Results

TERMS

undirected edge: an edge associated to a set $\{u, v\}$, where u and v are vertices

directed edge: an edge associated to an ordered pair (u, v), where u and v are vertices

multiple edges: distinct edges connecting the same vertices

multiple directed edges: distinct directed edges associated with the same ordered pair (u,v), where u and v are vertices

loop: an edge connecting a vertex with itself

undirected graph: a set of vertices and a set of undirected edges each of which is associated with a set of one or two of these vertices

simple graph: an undirected graph with no multiple edges or loops

multigraph: an undirected graph that may contain multiple edges but no loops

$\deg^-(v)$ (the in-degree of the vertex v in a graph with directed edges): the number of edges with v as their terminal vertex

$\deg^+(v)$ (the out-degree of the vertex v in a graph with directed edges): the number of edges with v as their initial vertex

underlying undirected graph of a graph with directed edges: the undirected graph obtained by ignoring the directions of the edges

K_n (complete graph on n vertices): the undirected graph with n vertices where each pair of vertices is connected by an edge

bipartite graph: a graph with vertex set that can be partitioned into subsets V_1 and V_2 so that each edge connects a vertex in V_1 and a vertex in V_2. The pair (V_1, V_2) is called a **bipartition** of V.

$K_{m,n}$ (complete bipartite graph): the graph with vertex set partitioned into a subset of m elements and a subset of n elements with two vertices connected by an edge if and only if one is in the first subset and the other is in the second subset

C_n (cycle of size n), $n \geq 3$: the graph with n vertices $v_1, v_2, \ldots, v_n$ and edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$

W_n (wheel of size n), $n \geq 3$: the graph obtained from C_n by adding a vertex and edges from this vertex to the original vertices in C_n

Q_n (n-cube), $n \geq 1$: the graph that has the 2^n bit strings of length n as its vertices and edges connecting every pair of bit strings that differ by exactly one bit

matching in a graph G: a set of edges such that no two edges have a common endpoint

complete matching M from V_1 to V_2: a matching such that every vertex in V_1 is an endpoint of an edge in M

maximum matching: a matching containing the most edges among all matchings in a graph

isolated vertex: a vertex of degree zero

pendant vertex: a vertex of degree one

regular graph: a graph where all vertices have the same degree

pseudograph: an undirected graph that may contain multiple edges and loops

directed graph: a set of vertices together with a set of directed edges each of which is associated with an ordered pair of vertices

directed multigraph: a graph with directed edges that may contain multiple directed edges

simple directed graph: a directed graph without loops or multiple directed edges

adjacent: two vertices are adjacent if there is an edge between them

incident: an edge is incident with a vertex if the vertex is an endpoint of that edge

deg v (degree of the vertex v in an undirected graph): the number of edges incident with v with loops counted twice

path from u to v in a graph with directed edges: a sequence of edges $e_1, e_2, \ldots, e_n$, where e_i is associated to (x_i, x_{i+1}) for $i = 0, 1, \ldots, n$, where $x_0 = u$ and $x_{n+1} = v$

simple path: a path that does not contain an edge more than once

circuit: a path of length $n \geq 1$ that begins and ends at the same vertex

connected graph: an undirected graph with the property that there is a path between every pair of vertices

cut vertex of G: a vertex v such that $G - v$ is disconnected

cut edge of G: an edge e such that $G - e$ is disconnected

nonseparable graph: a graph without a cut vertex

vertex cut of G: a subset V' of the set of vertices of G such that $G - V'$ is disconnected

$\kappa(G)$ (the vertex connectivity of G): the size of a smallest vertex cut of G

k-connected graph: a graph that has a vertex connectivity no smaller than k

edge cut of G: a set of edges E' of G such that $G - E'$ is disconnected

$\lambda(G)$ (the edge connectivity of G): the size of a smallest edge cut of G

connected component of a graph G: a maximal connected subgraph of G

strongly connected directed graph: a directed graph with the property that there is a directed path from every vertex to every vertex

strongly connected component of a directed graph G: a maximal strongly connected subgraph of G

Euler path: a path that contains every edge of a graph exactly once

Euler circuit: a circuit that contains every edge of a graph exactly once

Hamilton path: a path in a graph that passes through each vertex exactly once

Hamilton circuit: a circuit in a graph that passes through each vertex exactly once

weighted graph: a graph with numbers assigned to its edges

shortest-path problem: the problem of determining the path

subgraph of a graph $G = (V, E)$: a graph (W, F), where W is a subset of V and F is a subset of E

$G_1 \cup G_2$ (union of G_1 and G_2): the graph $(V_1 \cup V_2, E_1 \cup E_2)$, where $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$

adjacency matrix: a matrix representing a graph using the adjacency of vertices

incidence matrix: a matrix representing a graph using the incidence of edges and vertices

isomorphic simple graphs: the simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if there exists a one-to-one correspondence f from V_1 to V_2 such that $\{f(v_1), f(v_2)\} \in E_2$ if and only if $\{v_1, v_2\} \in E_1$ for all v_1 and v_2 in V_1

invariant for graph isomorphism: a property that isomorphic graphs either both have or both do not have

path from u to v in an undirected graph: a sequence of edges $e_1, e_2, \ldots, e_n$, where e_i is associated to $\{x_i, x_{i+1}\}$ for $i = 0, 1, \ldots, n$, where $x_0 = u$ and $x_{n+1} = v$

chromatic number: the minimum number of colors needed in a coloring of a graph

RESULTS

The handshaking theorem: If $G = (V, E)$ be an undirected graph with m edges, then $2m = \sum_{v \in V} \deg(v)$.

Hall's marriage theorem: The bipartite graph $G = (V, E)$ with bipartition (V_1, V_2) has a complete matching from V_1 to V_2 if and only if $|N(A)| \geq |A|$ for all subsets A of V_1.

There is an Euler circuit in a connected multigraph if and only if every vertex has even degree.

There is an Euler path in a connected multigraph if and only if at most two vertices have odd degree.

in a weighted graph such that the sum of the weights of the edges in this path is a minimum over all paths between specified vertices

traveling salesperson problem: the problem that asks for the circuit of shortest total length that visits every vertex of a weghted graph exactly once

planar graph: a graph that can be drawn in the plane with no crossings

regions of a representation of a planar graph: the regions the plane is divided into by the planar representation of the graph

elementary subdivision: the removal of an edge $\{u, v\}$ of an undirected graph and the addition of a new vertex w together with edges $\{u, w\}$ and $\{w, v\}$

homeomorphic: two undirected graphs are homeomorphic if they can be obtained from the same graph by a sequence of elementary subdivisions

graph coloring: an assignment of colors to the vertices of a graph so that no two adjacent vertices have the same color

Dijkstra's algorithm: a procedure for finding a shortest path between two vertices in a weighted graph (see Section 10.6).

Euler's formula: $r = e - v + 2$ where r, e, and v are the number of regions of a planar representation, the number of edges, and the number of vertices, respectively, of a connected planar graph.

Kuratowski's theorem: A graph is nonplanar if and only if it contains a subgraph homeomorphic to $K_{3,3}$ or K_5. (Proof beyond scope of this book.)

The four color theorem: Every planar graph can be colored using no more than four colors. (Proof far beyond the scope of this book!)

11. Trees

Key Terms and Results

TERMS

tree: a connected undirected graph with no simple circuits

forest: an undirected graph with no simple circuits

rooted tree: a directed graph with a specified vertex, called the root, such that there is a unique path to every other vertex from this root

subtree: a subgraph of a tree that is also a tree

descendant of a vertex v in a rooted tree: any vertex that has v as an ancestor

internal vertex: a vertex that has children

leaf: a vertex with no children

level of a vertex: the length of the path from the root to this vertex

height of a tree: the largest level of the vertices of a tree

m-ary tree: a tree with the property that every internal vertex has no more than m children

full m-ary tree: a tree with the property that every internal vertex has exactly m children

binary tree: an m-ary tree with $m = 2$ (each child may be designated as a left or a right child of its parent)

ordered tree: a tree in which the children of each internal vertex are linearly ordered

parent of v in a rooted tree: the vertex u such that (u, v) is an edge of the rooted tree

child of a vertex v in a rooted tree: any vertex with v as its parent

sibling of a vertex v in a rooted tree: a vertex with the same parent as v

ancestor of a vertex v in a rooted tree: any vertex on the path from the root to v

postorder traversal: a listing of the vertices of an ordered rooted tree defined recursively—the subtrees are listed in the order they occur from left to right, followed by the root

infix notation: the form of an expression (including a full set of parentheses) obtained from an inorder traversal of the binary tree representing this expression

prefix (or Polish) notation: the form of an expression obtained from a preorder traversal of the tree representing this expression

postfix (or reverse Polish) notation: the form of an expression obtained from a postorder traversal of the tree representing this expression

spanning tree: a tree containing all vertices of a graph

minimum spanning tree: a spanning tree with smallest possible sum of weights of its edges

balanced tree: a tree in which every leaf is at level h or $h-1$, where h is the height of the tree

binary search tree: a binary tree in which the vertices are labeled with items so that a label of a vertex is greater than the labels of all vertices in the left subtree of this vertex and is less than the labels of all vertices in the right subtree of this vertex

decision tree: a rooted tree where each vertex represents a possible outcome of a decision and the leaves represent the possible solutions of a problem

game tree: a rooted tree where vertices represents the possible positions of a game as it progresses and edges represent legal moves between these positions

prefix code: a code that has the property that the code of a character is never a prefix of the code of another character

minmax strategy: the strategy where the first player and second player move to positions represented by a child with maximum and minimum value, respectively

value of a vertex in a game tree: for a leaf, the payoff to the first player when the game terminates in the position represented by this leaf; for an internal vertex, the maximum or minimum of the values of its children, for an internal vertex at an even or odd level, respectively

tree traversal: a listing of the vertices of a tree

preorder traversal: a listing of the vertices of an ordered rooted tree defined recursively—the root is listed, followed by the first subtree, followed by the other subtrees in the order they occur from left to right

inorder traversal: a listing of the vertices of an ordered rooted tree defined recursively—the first subtree is listed, followed by the root, followed by the other subtrees in the order they occur from left to right

RESULTS

A graph is a tree if and only if there is a unique simple path between every pair of its vertices.

A tree with n vertices has $n-1$ edges.

A full m-ary tree with i internal vertices has $mi+1$ vertices.

The relationships among the numbers of vertices, leaves, and internal vertices in a full m-ary tree (see Theorem 4 in Section 11.1)

There are at most m^h leaves in an m-ary tree of height h.

If an m-ary tree has l leaves, its height h is at least $\lceil \log_m l \rceil$. If the tree is also full and balanced, then its height is $\lceil \log_m l \rceil$.

Huffman coding: a procedure for constructing an optimal binary code for a set of symbols, given the frequencies of these symbols

depth-first search, or backtracking: a procedure for constructing a spanning tree by adding edges that form a path until this is not possible, and then moving back up the path until a vertex is found where a new path can be formed

breadth-first search: a procedure for constructing a spanning tree that successively adds all edges incident to the last set of edges added, unless a simple circuit is formed

Prim's algorithm: a procedure for producing a minimum spanning tree in a weighted graph that successively adds edges with minimal weight among all edges incident to a vertex already in the tree so that no edge produces a simple circuit when it is added

Kruskal's algorithm: a procedure for producing a minimum spanning tree in a weighted graph that successively adds edges of least weight that are not already in the tree such that no edge produces a simple circuit when it is added

12. Boolean Algebra

Key Terms and Results

TERMS

Boolean variable: a variable that assumes only the values 0 and 1

$\overline{x}$ **(complement of x):** an expression with the value 1 when x has the value 0 and the value 0 when x has the value 1

$x \cdot y$ (or xy) **(Boolean product** or **conjunction of x and y):** an expression with the value 1 when both x and y have the value 1 and the value 0 otherwise

$x + y$ **(Boolean sum** or **disjunction of x and y):** an expression with the value 1 when either x or y, or both, has the value 1, and 0 otherwise

Boolean expressions: the expressions obtained recursively by specifying that $0, 1, x_1, \ldots, x_n$ are Boolean expressions and $\overline{E}_1$, $(E_1 + E_2)$, and $(E_1 E_2)$ are Boolean expressions if E_1 and E_2 are

dual of a Boolean expression: the expression obtained by interchanging $+$ signs and $\cdot$ signs and interchanging 0s and 1s

Boolean function of degree n: a function from B^n to B where $B = \{0, 1\}$

Boolean algebra: a set B with two binary operations $\vee$ and $\wedge$, elements 0 and 1, and a complementation operator $^-$ that satisfies the identity, complement, associative, commutative, and distributive laws

literal of the Boolean variable x: either x or $\overline{x}$

minterm of $x_1, x_2, \ldots, x_n$: a Boolean product $y_1 y_2 \cdots y_n$, where each y_i is either x_i or $\overline{x}_i$

$x \downarrow y$ **(or x *NOR* y):** the expression that has the value 0 when either x or y or both have the value 1 and the value 0 otherwise

inverter: a device that accepts the value of a Boolean variable as input and produces the complement of the input

OR gate: a device that accepts the values of two or more Boolean variables as input and produces their Boolean sum as output

AND gate: a device that accepts the values of two or more Boolean variables as input and produces their Boolean product as output

half adder: a circuit that adds two bits, producing a sum bit and a carry bit

full adder: a circuit that adds two bits and a carry, producing a sum bit and a carry bit

K-map for n variables: a rectangle divided into 2^n cells where each cell represents a minterm in the variables

minimization of a Boolean function: representing a Boolean function as the sum of the fewest products of literals such that these products contain the fewest literals possible among all sums of products that represent this Boolean function

implicant of a Boolean function: a product of literals with the property that if this product has the value 1, then the value of this Boolean function is 1

prime implicant of a Boolean function: a product of literals that is an implicant of the Boolean function and no product obtained by deleting a literal is also an implicant of this function

sum-of-products expansion (or **disjunctive normal form**): the representation of a Boolean function as a disjunction of minterms

functionally complete: a set of Boolean operators is called functionally complete if every Boolean function can be represented using these operators

$x \mid y$ (or x ***NAND*** y)**:** the expression that has the value 0 when both x and y have the value 1 and the value 1 otherwise

An identity between Boolean functions represented by Boolean expressions remains valid when the duals of both sides of the identity are taken.

Every Boolean function can be represented by a sum-of-products expansion.

essential prime implicant of a Boolean function: a prime implicant of the Boolean function that must be included in a minimization of this function

don't care condition: a combination of input values for a circuit that is not possible or never occurs

RESULTS

The identities for Boolean algebra (see Table 5 in Section 12.1).

Each of the sets $\{+, \bar{}\}$ and $\{\cdot, \bar{}\}$ is functionally complete.
Each of the sets $\{\downarrow\}$ and $\{\mid\}$ is functionally complete.
The use of K-maps to minimize Boolean expressions.
The Quine–McCluskey method for minimizing Boolean expressions.

13. Modeling Computation

Key Terms and Results

TERMS

alphabet (or vocabulary): a set that contains elements used to form strings

language: a subset of the set of all strings over an alphabet

phrase-structure grammar (V, T, S, P)**:** a description of a language containing an alphabet V, a set of terminal symbols T, a start symbol S, and a set of productions P

the production $w \rightarrow w_1$**:** w can be replaced by w_1 whenever it occurs in a string in the language

$w_1 \Rightarrow w_2$ **(**w_2 **is directly derivable from** w_1**):** w_2 can be obtained from w_1 using a production to replace a string in w_1 with another string

$w_1 \overset{*}{\Rightarrow} w_2$ **(**w_2 **is derivable from** w_1**):** w_2 can be obtained from w_1 using a sequence of productions to replace strings by other strings

type 0 grammar: any phrase-structure grammar

type 1 grammar: a phrase-structure grammar in which every production is of the form $w_1 \rightarrow w_2$, where $w_1 = lAr$ and $w_2 = lwr$, where $A \in N$, $l, r, w \in (N \cup T)*$ and $w \neq \lambda$, or $w_1 = S$ and $w_2 = \lambda$ as long as S is not on the right-hand side of another production

type 2, or context-free, grammar: a phrase-structure grammar in which every production is of the form $A \rightarrow w_1$, where A is a nonterminal symbol

type 3, or regular, grammar: a phrase-structure grammar where every production is of the form $A \rightarrow aB$, $A \rightarrow a$, or $S \rightarrow \lambda$, where A and B are nonterminal symbols, S is the start symbol, and a is a terminal symbol

derivation (or parse) tree: an ordered rooted tree where the root represents the starting symbol of a type 2 grammar, internal vertices represent nonterminals, leaves represent terminals, and the children of a vertex are the symbols on the right side of a production, in order from left to right, where the symbol represented by the parent is on the left-hand side

Backus–Naur form: a description of a context-free grammar in which all productions having the same nonterminal as their left-hand side are combined with the different right-hand sides of these productions, each separated by a bar, with nonterminal symbols enclosed in angular brackets and the symbol $\rightarrow$ replaced by ::=

finite-state machine (S, I, O, f, g, s_0) **(or a Mealy machine):** a six-tuple containing a set S of states, an input alphabet I, an output alphabet O, a transition function f that assigns a next state to every pair of a state and an input, an output function g that assigns an output to every pair of a state and an input, and a starting state s_0

AB **(concatenation of** A **and** B**):** the set of all strings formed by concatenating a string in A and a string in B in that order

A^* **(Kleene closure of** A**):** the set of all strings made up by concatenating arbitrarily many strings from A

deterministic finite-state automaton (S, I, f, s_0, F)**:** a five-tuple containing a set S of states, an input alphabet I, a transition function f that assigns a next state to every pair of a state and an input, a starting state s_0, and a set of final states F

nondeterministic finite-state automaton (S, I, f, s_0, F)**:** a five-tuple containing a set S of states, an input alphabet I, a transition function f that assigns a set of possible next states to every pair of a state and an input, a starting state s_0, and a set of final states F

language recognized by an automaton: the set of input strings that take the start state to a final state of the automaton

regular expression: an expression defined recursively by specifying that $\emptyset$, λ, and x, for all x in the input alphabet, are regular expressions, and that $(\mathbf{AB})$, $(\mathbf{A} \cup \mathbf{B})$, and $\mathbf{A}^*$ are regular expressions when $\mathbf{A}$ and $\mathbf{B}$ are regular expressions

regular set: a set defined by a regular expression (see page 820)

Turing machine $T = (S, I, f, s_0)$**:** a four-tuple consisting of a finite set S of states, an alphabet I containing the blank symbol B, a partial function f from $S \times I$ to $S \times I \times \{R, L\}$, and a starting state s_0

nondeterministic Turing machine: a Turing machine that may have more than one transition rule corresponding to each (state, tape symbol) pair

decision problem: a problem that asks whether statements from a particular class of statements are true

solvable problem: a problem with the property that there is an effective algorithm that can solve all instances of the problem

unsolvable problem: a problem with the property that no effective algorithm exists that can solve all instances of the problem

computable function: a function whose values can be computed using a Turing machine

uncomputable function: a function whose values cannot be computed using a Turing machine

P, the class of polynomial-time problems: the class of problems that can be solved by a deterministic Turing machine in polynomial time in terms of the size of the input

NP, the class of nondeterministic polynomial-time problems: the class of problems that can be solved by a nondeterministic Turing machine in polynomial time in terms of the size of the input

NP-complete: a subset of the class of NP problems with the property that if any one of them is in the class P, then all problems in NP are in the class P

RESULTS

For every nondeterministic finite-state automaton there is a deterministic finite-state automaton that recognizes the same set.

Kleene's theorem: A set is regular if and only if there is a finite-state automaton that recognizes it.

A set is regular if and only if it is generated by a regular grammar.

The halting problem is unsolvable.