Karp's 21 NP-complete problems:

- 0-1 integer programming
- clique
  - set packing (level denotes SAT $\rightarrow$ clique $\rightarrow$ set packing)
  - vertex cover
    - set covering
    - feedback node set
    - feedback arc set
    - directed Hamiltonian circuit
      - undirected Hamiltonial circuit
- 3-SAT
  - chromatic number
    - clique cover
    - exact cover
      - hitting set
      - Steiner tree
      - 3-dimensional matching
      - knapsack
        - job sequencing
        - patition
          - max cut

NP-complete satisfiability problems:

- MAX2SAT = $\{ <\phi, k> \mid \exists$ a truth assignment that satisfies at least k clauses of a 2cnf-formula $\phi\}$.
- MAXKSAT = $\{ <\phi, r> \mid \exists$ a truth assignment that satisfies at least r clauses of a k-cnf-formula $\phi\}$.
- NAESAT[1] = $\{<\phi> \mid \exists$ a truth assignment wherein for every clause $c_i \in \phi$ there exists at least one true literal and one false literal$\}$
- *MAXSAT*: $\{ <\phi, r> \mid \exists$ a truth assignment that satisfies at least r clauses of $\phi\}$.

[1] Not All Equal SAT

- *MINKSAT*: $\{ < \phi, r > \mid \exists$ a truth assignment that satisfies at most r clauses of a k-cnf-formula $\phi\}$.
- *Monotone 3SAT*: $\{ < \phi > \mid \exists$ a truth assignment that satisfies ϕ wherein all literals in every clause of ϕ are either postive or negative$\}$.
- *1-in-3SAT*: $\{ < \phi > \mid \exists$ a truth assignment that satisfies ϕ while setting exactly one literal to true in each clause of $\phi\}$.
- *CIRCUIT-SAT*: $\{< C > \mid C$ is a satisfiable boolean circuit$\}$.

- *Planar 3SAT*:
- *Planar rectilinear 3SAT*
- *Planar monotone rectilinear 3SAT*
- *Planar 1-in-3SAT*

Satisfiability problems that have polynomial-time algorithms:

- *2SAT*: each clause has at most two literals
- *Horn SAT*: each clause has ≤ 1 positive literals
- *Dual-horn SAT*: each clause has ≤ 1 negative literals
- *DNF SAT*: formula is $\vee$ of clauses; clause is $\wedge$ of literals
- *planar circuit SAT*: given a boolean circuit that can be embedded in the plane so that no two wires cross, is there an input that makes the circuit output TRUE?
- *planar NAE 3SAT*:

NP-complete graph-theoretic problems:

- *Dominating set*: Given a graph $G(V, E)$ and an integer parameter k, decide whether there exists a set $V' \subseteq V$ with $|V'| \leq k$ of vertices such that for every vertex not in V' has a neighbor in V'.
- *Independent set*: Given a graph $G(V, E)$ and an integer parameter k, decide whether there exists a set $V' \subseteq V$ with $|V'| \geq k$ of vertices such that each edge in E incident on at most one vertex in V'.
- *Max bisection*: Given an undirected graph $G(V, E)$ and an integer parameter k, decide whether there exists a vertex cut $(S, V - S)$ of size k or more such that $|S| = |V - S|$.
- *Bisection width*: Given an undirected graph $G(V, E)$ and an integer parameter k, decide whether there exists a vertex cut $(S, V - S)$ of size at most k such that $|S| = |V - S|$.

- *Feedback vertex set*: Given an undirected graph $G(V, E)$ and k, decide whether there exists $V' \subseteq V$ with $|V'| \leq k$ such that removing vertices in V' leaves G acyclic.

- *Feedback arc set*: Given a directed graph $G(V, E)$ and k, decide whether there exists $E' \subseteq E$ with $|E'| \leq k$ such that removing arcs in E' leaves G acyclic.[2]

- *HAM-CYCLE*: Decide whether the given directed/undirected graph G has a Hamiltonian cycle.

- *Traveling salesman tour*: Given an undirected/directed edge-weighted graph $G(V, E)$ and a parameter M, decide whether G has a Hamiltonian cycle of weight at most M.[3]

- *Multicommidity max-flow*: Given a directed gaph $D(V, A)$ and $2k$ nodes, $s_1, \ldots, s_k, t_1, \ldots t_k$ in V, are there node-disjoint directed paths from s_1 to t_1, s_2 to t_2, $\ldots$, and s_k to t_k?

- *Graph coloring*: Given a graph $G(V, E)$ and k, decide whether the vertices (resp. edges) of G can be colored using at most k colors.

- *Crossing number of G*

- *Steiner tree of G*

Set-theoretic NP-complete problems:

- *Set Packing*: Given a collection C of finite sets, deciding whether there exists a set packing i.e., a collection of disjoint sets $C' \subseteq C$ such that the cardinality of C' is k.

- *Set Cover*: Given a collection C of subsets of a finite set S, deciding whether there exists a set cover for X i.e., a subset $C' \subseteq C$ such that every element in S belongs to at least one member of C' and the cardinality of the set cover C' is at most k.

- *3-Exact Cover*: Given a family $F = \{S_1, \ldots, S_n\}$ of n subsets of $S = \{u_1, \ldots, u_{3m}\}$ each of cardinality three, is there a subfamily of m subsets that covers S?

- *Tripatrite (3-Dimensional) Matching*: Given three sets U, V, and W of equal cardinality, and a subset T of $U \times V \times W$, is there a subset M of T with $|M| = |U|$ such that whenever (u, v, w) and (u', v', w') are distinct triples with M, $u \neq u', v \neq v'$, and $w \neq w'$?

- *Hitting Set*: Given a collection C of subsets of a finite set S, deciding whether there exists a subset $S' \subseteq S$ such that S' contains at least one element from each subset in C s.t. the cardinality of S' is at most k.

- *Partition*: Given integers $c_1, \ldots, c_n$, is the subset $S \subseteq \{1, \ldots, n\}$ such that $\sum_{j \in S} c_j = \sum_{j \notin S} c_j$?

- *Integer Knapsack*: Given integers c_j, $j = 1, \ldots, n$ and k are there integers $x_j \geq 0$, $j = 1, \ldots, n$ such that $\sum_{j=1}^{n} c_j x_j = k$?

- 0-1 *Knapsack*: Given integers c_j, $j = 1, \ldots, n$ and k, is there subset S of $\{1, \ldots, n\}$ such that $\sum_{j \in S} c_j = k$?

[2] undirected graphs version $\in P$

[3] NP-complete in Euclidan plane as well

- *Bin Packing*: Given a finite set U of items, a size $s(u) \in Z^+$ for each $u \in U$, and a positive integer bin capacity B, finding a partition of U into disjoint sets $U_1, U_2, \ldots, U_m$ such that the sum of the items in each U_i is B or less and the number of used bins (i.e, the number of disjoint sets) m is at most k.

Miscellaneous NP-complete problems:

- *Multiprocessor Scheduling*: Given a set T of tasks, number m of processors, length $l(t,i) \in Z^+$ for each task $t \in T$ and processor $i \in [1, \ldots, m]$, deciding whether there exists an m-processor schedule for T i.e., a function $f : T \to [1 \ldots m]$ s.t. the finish time for the schedule is at most t.
- *Job Shop Scheduling*: Given n jobs of varying sizes, deciding whether there exists a schedule of these jobs on m identical machines, such that the total length of the schedule is at most t.
- *Integer linear programming*: Linear program in which each variable is restricted to be an integer, deciding whether there exists a solution such that the minimizing (resp. maximizing) objective function value is at most (resp. at least) k.
- *Quadratic programming*:

NP-complete	P
3SAT	2SAT, HORN-SAT [4]
traveling salesman problem	minimum spanning tree
minimum Steiner tree	minimum spanning tree
vertex cover	edge cover
longest path in graphs	shortest path in graphs
maximum cut	minimum cut
tripartite matching	bipartite matching
indepdent set on graphs	independent set on trees
Hamiltonian path	Eulerian path
integer programming	linear programming[5]
k-colorability with $k > 2$	2-colorability
feedback arc set	feedback edge set

- Euclidean shortest paths in $\mathbb{R}^2 \in P$

  Euclidean shortest paths in $\mathbb{R}^3 \in NP$-hard but no proof showing that it belongs to NP: Given a polyhedral domain, two points $s, t \in \mathbb{R}^3$ and k, determining whether there exists a shortest path between s and t that is of distance at most k.

- graph isomorphism $\in NPI$ (in NP, but no proof to show that it belongs to either P or NP-complete)

  subgraph isomorphism $\in NP$-complete

- The primality problem: Given any positive integer n in binary, determine whether n is a prime.

  -For this problem to belong to class P, there must exist an algorithm that takes $O((\lg n)^k)$ time for some constant k.

  -primality $\in P$ (due to primality testing algorithm by AKS)

- The composite numbers (aka, positive integer factorization) problem: Given any positive integer n in binary, determine whether there are integers $n', n'' > 1$ such that $n = n'n''$.

  -For this problem to belong to class P, there must exist an algorithm that takes $O((\lg n)^k)$ time for some constant k.

  -integer factorization $\in NP \cap coNP$ but no proof showing that it belongs to either class P or class NP-hard