Algorithmic Problem Definitions

(NB: all of these problems are NP-hard)

(Max-)Clique. Input is a graph. Output is a clique (subset of vertices wherein all possible edges appear). Value is the *fraction* of vertices in the clique.

(Min-)Coloring. Input is a graph. Output is a legal coloring (all edges bichromatic). Value is the number of colors used.

Max-Coverage. Input is the same as in Set-Cover, plus a positive integer m. Output is a subcollection of exactly m sets. Value is the *fraction* of ground elements covered by the subcollection.

Max-Cut. Input is a graph. Output is a 2-coloring. Value is the fraction of edges that are "cut"; i.e., bichromatic.

Max-kCut. Same as Max-Cut except that the output is a k-coloring.

(Min-)Hitting-Set. Input is the same as in Set-Cover. Output is a hitting set (subset of vertices such that all sets contain at least one vertex). Value is the fraction of vertices in the hitting set.

(Max-)Independent-Set. Input is a graph. Output is an independent set (subset of vertices with no edges between them). Value is the fraction of vertices in the independent set.

(Max-)Label-Cover(K,L). Input is a bipartite graph with left vertices U, right vertices V, and edges E. Also part of the input are "(projection) constraints": for each edge (u, v) this is an explicitly written function $\pi_{v\to u} : L \to K$. Here L and K are "constant-size" sets that are not part of the input. (Think of $|L| \geq |K|$ and perhaps $|V| \geq |U|$.) Edges and constraints are usually identified. Output is a "labeling"/"assignment": a map $f : U \to K, V \to L$. Value is the fraction of constraints "satisfied" where we say f satisfies a constraint $\pi_{v\to u}$ if $f(u) = \pi_{v\to u}(f(v))$.

(Max-)Unique-Label-Cover(L). Same as Label-Cover except that $K = L$ and all constraints $\pi_{v\to u}$ are actually *bijections* (permutations) on L.

(Max-)k-ary-Consistent-Labeling(K,L). Somewhat similar to Label-Cover. Input is basically a k-uniform hypergraph $H = (V, E)$, except the "hyperedges" are not sets of k vertices but rather ordered lists of vertices (duplicates allowed) of length k; here $k \geq 2$ is an integer. Also part of the input is a "constraint" for each "hyperedge" $e = (v_1, \dots, v_k)$; this is list of maps $\pi_e^1, \dots, \pi_e^k : L \to K$. Output is a map $f : V \to L$. Given a constraint, consider the list of keys $\pi_e^1(f(v_1)), \dots, \pi_e^k(f(v_k))$. If all keys are the same we say f "strongly satisfies" e; if at least two are the same we say f "weakly satisfies" e; if all keys are different we say f "violates" e. The "strong (resp. weak) value" of f is the fraction of constraints it strongly (resp. weakly) satisfies.

Max-3Lin. Input is a collection of linear equations mod 2 over n variables, each equation involving at most 3 variables. Output is an assignment of integers mod 2. Value is the fraction of equations satisfied.

Max-kLin. Same as Max-3Lin except each equation involves at most k variables.

Max-EkLin. Same as Max-kLin except each equation involves *exactly* k variables.

Max-kLin(q). Same as Max-kLin except the variables and equations are modulo q.

Max-3Sat. Input is a CNF formula over n variables in which each clause has *at most* 3 literals. Output is a 0-1 assignment to the variables. Value is the fraction of clauses satisfied.

Max-Sat. Same as Max-3Sat except there is no restriction on the number of literals per clause.

Max-kSat. Same as Max-3Sat except each clause has at most k literals.

Max-EkSat. Same as Max-kSat except each clause has *exactly* k literals.

Max-kSat-b. Same as Max-kSat with the additional guarantee that each variable appears in exactly b clauses.

(Min-)Set-Cover. Input is a collection of M nonempty sets of "ground elements". n is the total number of ground elements (in the union of all the sets). Output is a covering subcollection of sets (one whose union is all of the ground elements). Value is the fraction of sets in the covering subcollection.

(Min-)TSP. Input is an undirected complete graph in which each edge has a "distance" in the range $[0, \infty]$. Output is a tour (cyclic path) visiting each vertex exactly once. Value is the total distance of the tour.

(Min-)Metric-TSP. Same as TSP except the distances are constrained to satisfy the "triangle inequality": $\text{dist}(u, w) \leq \text{dist}(u, v) + \text{dist}(v, w)$ for all vertices u, v, w.

(Min-)Euclidean-TSP. Same as TSP except the vertices are required to lie in $\mathbb{R}^n$ and the distances are the usual Euclidean distances.

(Min-)$\mathbb{R}^d$-TSP. Same as Euclidean-TSP except the vertices are required to lie in $\mathbb{R}^d$.

(Min-)Vertex-Cover. Input is a graph. Output is a vertex cover (subset of vertices such that every edge includes one of the vertices). Value is the fraction of vertices in the cover.

(Min-)Ek-Vertex-Cover. Same as Vertex-Cover except that the input is a k-uniform hypergraph.