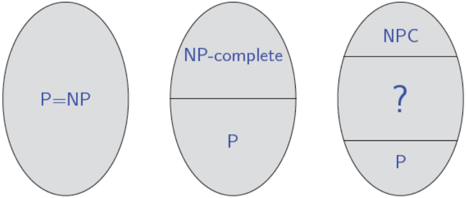
Decision vs. Search Problems

NP-Intermediate Problem

- For $P \stackrel{?}{=} NP$ question, which of the following three is most likely?



Theorem 1 (Ladner 1975)

If $P \neq NP$ then there is a language in NP that is neither in P not NP-complete.

- Ladner's theorem gives an "artificial" problem between P and NP.
- Possible candidates include Graph Isomorphism, Total search problems (Factoring, Nash equilibrium computation, and others)

Decision vs. Search Problem

- For NP-complete problem SAT, suppose we want to compute a satisfying assignment, not just test for satisfiability
- Given a *SAT*-oracle, do the following.
  For ϕ over variables $x_1, ..., x_n$, check if ϕ is satisfiable, if so, try ϕ with x_1 assigned to 0; otherwise, assign 1 to x_1, then proceed to x_2 etc.
- In a sense, computing a satisfying assignment (a search problem) is no harder than checking SAT (a decision problem).
- Subsequently, we shall see Complexity class *FNP*: functions checkable in poly-time.
  - FSAT is FNP-complete
  - HOw about function versions of other NP-complete problems?

Polynomial-time Checkable Relations

- **FNP**: *NP search problem* (a.k.a. function computation problem) is a binary relation $R(.,.)$ such that
  - $R(x,y)$ is checkable in time polynomial in $|x|$ and $|y|$
  - given input x, the goal is to find y with $R(x,y)$ (y is regarded as a certificate)
- **TFNP**: *Total search problem:* Search problem defined for all inputs, i.e.,

  $$\forall x \exists y(|y| = poly(|x|), R(x,y))$$

  Examples include
  - Local-max-cut:
  - Factoring: input a number N, output the prime factorization of N.
  - Nash: the problem of computing a Nash equilibrium of a game
  - ...
- **FP**: there exists a det. Ptime algorithm solving it. So given input x, it returns a solution y so that $R(x,y)$ or it states such an y does not exist.

Definition 2

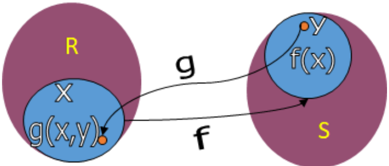
Let R and S be search problems in FNP. We say that R is many-one reducible to S, if there exist polynomial-time computable functions f, g such that

$$(f(x), y) \in S \Rightarrow (x, g(x, y)) \in R$$

.

Theorem 3

FSAT (the problem of finding a satisfying assignment of a boolean formula) is FNP-complete.



Reducibility for Search Problems

- If S is polynomial-time solvable, then so is R.
- Two problems R and S are (polynomial-time) equivalent, if R reduces to S and S reduces to R.
- SAT: x is boolean formula, y is satisfying assignment. Decision version of SAT is polynomial-time equivalent to search for y.

Theorem 4

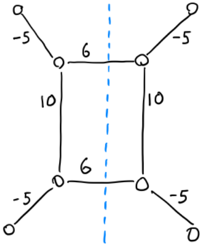
There is an FNP-complete problem in TFNP if and only if $NP = co - NP$.

- Factoring (for example) cannot be NP-hard unless $NP = co - NP$. Unlikely! So Factoring is in strong sense "NP-intermediate".
- Other such problems include: Local-Max-Cut, NASH, ...

Polynomial Local Search (PLS)

- **Max-Cut**: Find a cut in a weighted graph G of maximum size. Decision version is a known NP-complete problem
- Note that **Min-Cut** is solvable in polynomial time.

Algorithm
$[A, B] \leftarrow$ an arbitrary partition of V
While placing some node v to the other side increases the cut weight,
    move v to the other side
return [A, B]



Generic Local Search Algorithm

- Move step by step from current solution to a "nearby" one

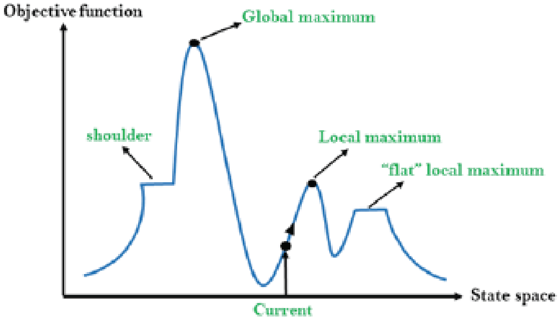
Algorithm

$s \leftarrow$ some initial solution

While a solution s' in the neighborhood of s is better than s: do

  $s \leftarrow s'$

return s



Key Characteristics of Local Search Algorithm

- Unlike greedy algorithms, we do not require maintaining a feasible solution all the time
  - Greedy algorithm typically build the solution bottom-up
- We need to have a solution so we can compute its value in order to determine whether or not to make a move to a neighboring solution
- Easy to design an algorithm
- Generally, No provable guarantees on the quality of the solution
- Can get a local optimum instead of a global optimum
- The larger the neighborhood, the better the resulting solution and the higher the running time

Theorem 5

$FP \subseteq PLS \subseteq TFNP \subseteq FNP$.

- Other Local Search Algorithm: Gradient Descent, Simulated Annealing, ...