# $\mathbf{coNP}$ problems

- $\mathbf{NP}$ is the class of problems with succinct certificates.
- $\mathbf{coNP}$ is the class of problems with succinct disqualifications.
- VALIDITY:
  INSTANCE: a Boolean expression $\phi$
  QUESTION: is $\phi$ valid?
- VALIDITY is in $\mathbf{coNP}$: for a expression $\phi$ which is not valid, a falsifying truth assignment is a succinct disqualification.
- HAMILTON PATH COMPLEMENT and SAT COMPLEMENT are also in $\mathbf{coNP}$.
- $\mathbf{P} \subseteq \mathbf{coNP}$





## $\mathbf{coNP}$-completeness

- HAMILTON PATH COMPLEMENT and VALIDITY are $\mathbf{coNP}$-complete:

  Every language $L \in \mathbf{coNP}$ is reducible to VALIDITY, because $\overline{L} \in \mathbf{NP}$ and, hence, there is a reduction $R$ from $\overline{L}$ to SAT such that for every string $x$, $x \in \overline{L}$ iff $R(x) \in$ SAT but then $x \in L$ iff $R(x) \notin$ SAT iff $\neg R(x) \in$ VALIDITY.

  Similarly for HAMILTON PATH COMPLEMENT.

**Proposition.** If $L$ is $\mathbf{NP}$-complete, then its complement $\overline{L} = \Sigma^* - L$ is $\mathbf{coNP}$-complete.





## $\mathbf{coNP}$ and $\mathbf{NP}$

- It is open whether $\mathbf{NP} = \mathbf{coNP}$.
- If $\mathbf{P} = \mathbf{NP}$, then $\mathbf{NP} = \mathbf{coNP}$.
- It is possible that $\mathbf{P} \neq \mathbf{NP}$ but $\mathbf{NP} = \mathbf{coNP}$ (however, it is strongly believed that $\mathbf{NP} \neq \mathbf{coNP}$).
- The problems in $\mathbf{coNP}$ that are $\mathbf{coNP}$-complete are the least likely to be in $\mathbf{P}$ and also in $\mathbf{NP}$ (see below).





## $\mathbf{coNP}$ and $\mathbf{NP}$—cont'd

**Proposition.** If a $\mathbf{coNP}$-complete problem is in $\mathbf{NP}$, $\mathbf{NP} = \mathbf{coNP}$.

Proof. Let $L$ be a $\mathbf{coNP}$-complete problem that is in $\mathbf{NP}$. Consider $L' \in \mathbf{coNP}$. Then there is a reduction $R$ from $L'$ to $L$. Then $L' \in \mathbf{NP}$, because $L'$ can be decided by a polynomial time NTM which on input $x$ computes first $R(x)$ and then starts the NTM for $L$. Thus $\mathbf{coNP} \subseteq \mathbf{NP}$.

Consider $L' \in \mathbf{NP}$. Then $\overline{L'} \in \mathbf{coNP}$ and there is a reduction $R$ from $\overline{L'}$ to $L$. Then $\overline{L'} \in \mathbf{NP}$ and hence $L' \in \mathbf{coNP}$. Thus $\mathbf{NP} \subseteq \mathbf{coNP}$.

## PRIMES

- PRIMES:
  INSTANCE: an integer $N$ in binary
  QUESTION: Is $N$ a prime number?
- PRIMES $\in \mathbf{coNP}$ (disqualification: divisor).
- Note that a $\mathcal{O}(\sqrt{N})$ algorithm for PRIMES testing all relevant divisor candidates is only pseudopolynomial.
- PRIMES $\in \mathbf{NP}$ (to be shown)
- August 2002:
  M. Agrawal, N. Kayal, N. Saxena: *PRIMES is in* $\mathbf{P}$ !!





## The class $\mathbf{coNP} \cap \mathbf{NP}$

- Problems in $\mathbf{coNP} \cap \mathbf{NP}$ have succinct certificates and disqualifications.
- $\mathbf{P} \subseteq \mathbf{coNP} \cap \mathbf{NP}$
- Certain kind of optimization problems are in $\mathbf{coNP} \cap \mathbf{NP}$:
  if two problems in $\mathbf{NP}$ are *dual*: each is *reducible to the complement* of the other, then both are in $\mathbf{coNP} \cap \mathbf{NP}$.





## The class $\mathbf{coNP} \cap \mathbf{NP}$—cont'd

**Example.**
MAX FLOW(D): does a given network $N$ have a flow from $s$ to $t$ of at least $K$.
MIN CUT(D): Given a network and a budget $B$, is there are set of edges of capacity of at most $B$ such that deleting these disconnects $s$ from $t$.

Now by max flow–min cut theorem, $N$ *has* a flow of value at least $K$ iff it *does not have* a cut of capacity $K - 1$ or less.





## PRIMES—cont'd

A succinct certificate for primality can be obtained using the following theorem.

**Theorem.** A number $p > 1$ is prime iff there is a number $1 < r < p$ such that $r^{p-1} = 1 \bmod p$ and, furthermore, $r^{\frac{p-1}{q}} \neq 1 \bmod p$ for all prime divisors $q$ of $p - 1$.

**Corollary.** PRIMES is in $\mathbf{NP} \cap \mathbf{coNP}$.

## PRIMES—cont'd

Proof.

- The succinct certificate $C(p)$ for primality of $p$ is
  $C(p) = (r; q_1, C(q_1), \ldots, q_k, C(q_k))$
  where $C(q_i)$ is a primality certificate for $q_i$.
- The certificate is polynomial in the length of $p$ (in $\log p$).
- The certificate can be checked by division and exponentiation.
- Ordinary multiplication and division doable in polynomial time in the length of the input (in binary).
- Exponentiation $r^{p-1}$ can be done in polynomial time by starting with repeated squaring $r^2, r^4, \ldots r^{2^l}$.
- Hence, the certificate can be checked in polynomial time.





## Function problems

- We have studied decision problems but many problems in practice require a more complicated answer than "yes"/"no".
  **Example.** Find a satisfying truth assignment for a formula. Compute an optimal tour for TSP.
- Such problems are called *function problems*.
- Decision problems are useful surrogates of function problems only in the context of *negative complexity results*.
  **Example.** SAT and TSP(D) are $\mathbf{NP}$-complete. Then unless $\mathbf{P} = \mathbf{NP}$, there is no polynomial time algorithm for finding a satisfying truth assignment or an optimal tour.





## Function problems—cont'd

- FSAT: given a Boolean expression $\phi$, if $\phi$ is satisfiable then return a satisfying truth assignment of $\phi$ otherwise return "no".
- If FSAT can solved in polynomial time, then so can SAT.
- If SAT can solved in polynomial time, then so can FSAT using the following algorithm given $\phi$ with variables $x_1, \ldots, x_n$ ($\phi[x := \mathbf{true}]$ denotes $\phi$ where variable $x$ is replaced by $\mathbf{true}$):

  if $\phi \notin$ SAT then return "no";
  for all $x \in \{x_1, \ldots, x_n\}$ do
    if $\phi[x := \mathbf{true}] \in$ SAT then $T(x) := \mathbf{true}$; $\phi := \phi[x := \mathbf{true}]$
    else $T(x) := \mathbf{false}$ ; $\phi := \phi[x := \mathbf{false}]$;
  return $T$;





## Function problems—cont'd

- If TSP can solved in polynomial time, then so can TSP(D).
- If TSP(D) can solved in polynomial time, then so can TSP.
- An optimal tour can be found using an algorithm which
  1. finds the cost $C$ of an optimal tour by binary search (with a polynomial number of calls to the polynomial time TSP(D));
  2. finds an optimal tour using $C$ (with a polynomial number of calls to the polynomial time TSP(D)).

A TSP algorithm using a TSP(D) algorithm as a subroutine:

/*Find the cost $C$ of an optimal tour by binary search*/<br>
$C := 0$; $C_u := 2^n$;<br>
while $(C_u > C)$ do<br>
  if there is a tour of cost $\lfloor (C_u + C)/2 \rfloor$ or less then<br>
    $C_u := \lfloor (C_u + C)/2 \rfloor$<br>
  else $C := \lfloor (C_u + C)/2 \rfloor + 1$;<br>
/* Find an optimal tour */<br>
For all intercity distances do<br>
  set the distance to $C + 1$;<br>
  if there is a tour of cost $C$ or less, freeze the distance to $C + 1$<br>
  else restore the original distance and add it to the tour;<br>
endfor





## Relationship between decision and function problems

- Let $L$ be a language in $\mathbf{NP}$ having a polynomial time decidable and polynomially balanced relation $R_L$ such that for all strings $x$: there is a string $y$ with $R_L(x, y)$ iff $x \in L$.
- The function problem associated with $L$, denoted by F$L$:
  Given $x$, find a string $y$ such that $R_L(x, y)$ if such a string $y$ exists; if no such string exists, return "no".
- The class of all function problems associated as above with languages in $\mathbf{NP}$ is called $\mathbf{FNP}$.
- $\mathbf{FP}$ is the subclass of $\mathbf{FNP}$ solvable in polynomial time.
- FSAT is in $\mathbf{FNP}$ and FHORNSAT is in $\mathbf{FP}$
  (but it is open whether TSP is in $\mathbf{FNP}$).





## Reductions of function problems

- A function problem $A$ *reduces* to a function problem $B$ if there are string functions $R$, $S$ computable in logarithmic space such that for all strings $x, z$: if $x$ is an instance of $A$, then $R(x)$ is an instance of $B$ and if $z$ is a correct output of $R(x)$, then $S(z)$ is a correct output of $x$.
- A problem $A$ is *complete* for a class F$\mathcal{C}$ of function problems if it is in F$\mathcal{C}$ and every problem in F$\mathcal{C}$ reduces to $A$.
- $\mathbf{FP}$ and $\mathbf{FNP}$ are closed under reductions and reductions among function problems compose.
- FSAT is $\mathbf{FNP}$-complete.
- $\mathbf{FP} = \mathbf{FNP}$ iff $\mathbf{P} = \mathbf{NP}$.





## Total functions

- There are certain important problems in $\mathbf{FNP}$ that are guaranteed to never return "no".
- FACTORING: Given an integer $N$, find its prime decomposition
  $N = p_1^{k_1} \cdots p_m^{k_m}$
  (No known polynomial time algorithm).
- FACTORING seems to be different from the other hard problems in $\mathbf{FNP}$: it is a total function.
- A problem $L$ in $\mathbf{FNP}$ is called *total* if for every string $x$ there is at least one string $y$ such that $R_L(x, y)$.
- The subclass of $\mathbf{FNP}$ containing all total function problems is denoted by $\mathbf{TFNP}$.

## Total functions—cont'd

There are also other problems in **TFNP** with no known polynomial time algorithm.

**Example.** HAPPYNET:
INSTANCE: an undirected graph $(V, E)$ with integer weights $w$ on edges.
QUESTION: find a state of the graph where all nodes are happy.

- A state is a mapping $S : V \mapsto \{-1, +1\}$
- A node $i$ is happy in S if $S(i) \sum_{[i,j] \in E} S(j) w[i, j] \geq 0$
- Every graph has a happy state:
  A happy state can be found using the algorithm:
  Start with any S and while there is an unhappy node, flip it.





## HAPPYNET—cont'd

- This algorithm in not polynomial but pseudopolynomial $\mathcal{O}(W)$ where $W$ is the sum of all weights.
- No polynomial algorithm known.
- HAPPYNET equivalent with finding stable states in neural networks in the Hopfield model.





## Total functions—cont'd

- ANOTHER HAMILTON CYCLE is **FNP**-complete.
- ANOTHER HAMILTON CYCLE for cubic graphs is in **TFNP**.
- EQUAL SUMS:
  Given $n$ positive integers $a_1, \ldots, a_n$ such that $\sum_{i=1}^{n} a_i < 2^n - 1$, find two different subsets that have the same sum.
- EQUAL SUMS in **TFNP**.