Complexity classes

- **P** – class of *decision* problems that can be solved by a polynomial-time algorithm
- **NP** – non-deterministic polynomial time
  - Given a solution, it can be checked in polynomial time
    - given a cycle/tour – check?
    - Composite number: given a factor easy to check (but to find one?)
Theorems, conjectures

- **Theorem**: $P \subseteq NP$
  - If we can solve a problem then we can surely check it

- The central open question:
  - Is $P=NP$ or not?

- **Conjecture**: $P \neq NP$
  - Look at the hardest problems in $NP$
    - Harder than any other problem in $NP$

Polynomial reductions

- Two problems $A$ and $B$
- $A \leq^p B$:
  - $A$ is polynomially reducible to $B$
  - There is an algorithm for solving $A$ in time that would be polynomial if we could solve arbitrary instances of $B$ in unit time
  - If both ways then they are polynomially equivalent: $A \equiv^p B$
  - Transitive: if $A \leq^p B$ and $B \leq^p C$ then $A \leq^p C$
Example

- HAM \equiv^p \text{HAMD}
  - HAMD \leq^p \text{HAM}
    - Trivial: if HAMalgo finds a cycle then yes
  - HAM \leq^p \text{HAMD}
    - First check if HAMDalgo gives yes for the original graph
    - Start considering edges for removal one by one
      - Apply HAMDalgo to the remaining
      - If still Hamiltonian without an edge then remove it
      - Otherwise remove the edge and keep going
      - Stop when a cycle is left, return it

Reduction function

- Two decision problems \(X \subseteq I\) and \(Y \subseteq J\)
- F: map \(I \rightarrow J\)
  such that \(F(x) \in Y\) if and only if \(x \in X\)

Theorem: If F is computable in polynomial time then \(X \leq^p Y\)

```cpp
    bool DecideX(x) {
      y = F(x);
      if(DecideY(y)) return true;
      else return false;
    }
```
Example

- HAMD ≤ₚ TSPD
  - Given a graph G=(N,A), need to see if it is Hamiltonian
  - Define F(G) be the TSPD instance with a complete graph (N,N×N)
    - Cost = 1 if the edge in A and 2 otherwise
    - TSPD bound being N
    - If TSPD yes then that is a Hamiltonian cycle
    - If TSPD no then no Hamiltonian cycle

NP-completeness

- Decision problem X is NP-complete
  1. X is in NP
  2. Y ≤ₚ X for every problem Y in NP
- X is polynomially harder than any other NP problem
- If we know that X is NP-complete and X ≤ₚ Z then Z is NP-complete
- If we could only find one such X
SAT: satisfiability

- Given a boolean formula
  - Is it satisfiable? is there an assignment of values to variables that will make it true?
    - e.g. \((p \land q) \Rightarrow (p \lor q)\) is satisfiable via \((p = q = \text{true})\)
    - CNF: conjunctive normal form
  - SAT-CNF satisfiability for boolean expressions in CNF form
  - Cook’s Theorem
    - For any NP problem \(Y\), \(Y \leq^p \text{SAT-CNF}\)

NP-completeness

- Now that we know one of NP-complete problems
  - We find a bunch of other ones
  - All we need to show is
    - that a problem \(X\) is in NP
    - and that SAT-CNF \(\leq^p X\)
      - i.e. knowing that a polynomial algorithm for \(X\) would also mean that a polynomial algorithm exists for SAT-CNF
    - Then \(X\) is NP-complete
Some NP-complete problems

- SAT
- 3SAT: clauses have three variables
- 3DM: 3D matching
- HAMD: hamiltonian circuit
- PARTITION: set A and $s:A \rightarrow \mathbb{Z}^+$
  - Partition A into two equally sized parts
- CLIQUE: clique of size J or more
- VERTEX COVER: of size K or less
- K-COL: graph colorability with K colors or less

SAT-3-CNF is NP-complete

- It is in NP
- Will prove SAT-CNF $\leq_P$ SAT-3-CNF
  - Given an instance of SAT-CNF construct an instance of SAT-3-CNF efficiently
  Consider one clause: disjunction of K literals
  K=3: done
  K=4: $C = (L1 + L2 + L3 + L4)$ is satisfiable if and only if
  $C' = (L1 + L2 + U)(\neg U + L3 + L4)$ is satisfiable
  if C is true then $C'$ is satisfiable,
  if C is false then no choice of U will make $C'$ true.
SAT-3-CNF

- When $K>4$: $C=(L[1]+\ldots+L[K])$
  

  Again $C$ is true iff $C'$ is satisfiable

- Many clauses
  - Each with its own $U$'s
    - For an instance of SAT-CNF built an instance of SAT-3-CNF

SAT-2-CNF

- Is in P
  - Formula $((\neg X+Y)(\neg Z+\neg Y)(\neg Q+Z)(Q+\neg X))$
    - Construct directed graph: two verts for each variable, edge from $X$ to $Y$ if there is a clause equiv to $(\neg X+Y) = (\neg Y) + (\neg X)$
SAT-2-CNF

Claim

- If there are paths from some $N$ to $\neg N$ and from $\neg N$ to $N$ then the formula cannot be satisfied
- Otherwise it is satisfiable (example)

$$\neg X \rightarrow Y \quad Y \rightarrow \neg Z \quad \neg Z \rightarrow \neg Q \quad \neg Q \rightarrow \neg X$$

Hence $X \Rightarrow \neg X$

Hmm…

HAM vs Eulerian

- Hamiltonian path
  - directed graph $G=(V,E)$ and two vertices $s,t \in V$
  - decide if there exists a path from $s$ to $t$, which goes through each node once.
    - NP-complete (can construct graph for a SAT-3-CNF instance)
- Eulerian path
  - undirected graph $G=(V,E)$ and two vertices $s \neq t \in V$
  - decide if there exists a path from $s$ to $t$, which goes through each edge exactly once.
A theorem

Theorem: A connected graph has an Eulerian path from s to t iff
1. s and t’s degrees are odd.
2. the degrees of the other vertices are even
   So EULER is in P.

Can construct path by a polynomial algorithm?

NP-hard

- X is NP-hard
  - if there is an NP-complete problem Y that can be polynomially reduced to X
    • Y ≤p X
  - Does not have to be a decision problem
  - Decision problem can be NP-hard but not in NP, for instance exact K-colorability
    • Any K-coloring is a certificate for K-COL but not for K-COLE(exact: can color with K but no less)
What about two colors?

- Determine whether a graph is 2-vertex colorable
  - A polynomial algorithm?
    - DFS/BFS?

Metric TSP

- Undirected graph (complete)
- Distance matrix satisfies
  - Triangle inequality
    - $d(x,z) \leq d(x,y) + d(y,z)$
    - $\text{length(Hamiltonian cycle)} \geq \text{length(Hamiltonian path)} \geq \text{length(MST)}$
  - Construct an MST, tour around it will cost no more than $2*\text{length(MST)}$
  - tour with shortcuts $\leq 2 \times \text{length(MST)}$