Polynomial-Time Reduction

Reduction. Problem X polynomial reduces to problem Y if arbitrary instances of problem X can be solved using:
- Polynomial number of standard computational steps, plus
- Polynomial number (usually one!) call to oracle that solves problem Y.

Notation. $X \leq_p Y$.

Usually:
- Transform X instance into Y instance (in poly-time).
- Solve Y instance using oracle.
- Show that the oracle gives the answer to your X instance!
  i.e. "yes" for Y implies "yes" for X,
  and "yes" for X implies "yes" for Y (equiv to "no" Y implies "no" X).
NP-Complete

**NP-complete.** A problem such that

- $\exists Y \in \text{NP}$
- for every problem $X$ in $\text{NP}$, $X \leq_p Y$.

**Theorem.** Suppose $Y$ is an NP-complete problem. Then $Y$ is solvable in poly-time iff $P = \text{NP}$.

**Remark.** The condition that $Y \in \text{NP}$ is important.

---

Establishing NP-Completeness

**Recipe to establish NP-completeness of problem $Y$.**

- Step 1. Show that $Y$ is in $\text{NP}$.
- Step 2. Choose an $\text{NP}$-complete problem $X$, and prove that $X \leq_p Y$.

**Justification.** If $X$ is an $\text{NP}$-complete problem, and $Y$ is a problem in $\text{NP}$ with the property that $X \leq_p Y$ then $Y$ is $\text{NP}$-complete.
We will show: All problems below are NP-complete and polynomial reduce to one another!

Some NP-Complete Problems

Six basic genres of NP-complete problems and paradigmatic examples.
- Packing problems: SET-PACKING, INDEPENDENT SET.
- Covering problems: SET-COVER, VERTEX-COVER.
- Constraint satisfaction problems: SAT, 3-SAT.
- Sequencing problems: HAMILTONIAN-CYCLE, TSP.
- Partitioning problems: 3D-MATCHING, 3-COLOR.
- Numerical problems: SUBSET-SUM, KNAPSACK.

Practice. Most NP problems are either known to be in P or NP-complete.

Notable exceptions. Factoring, graph isomorphism, Nash equilibrium.
8.5 Sequencing Problems

Basic genres.
- Constraint satisfaction problems: SAT, 3-SAT.
- Packing problems: SET-PACKING, INDEPENDENT SET.
- Covering problems: SET-COVER, VERTEX-COVER.
- Sequencing problems: HAMILTONIAN-CYCLE, TSP.
- Partitioning problems: 3D-MATCHING, 3-COLOR.
- Numerical problems: SUBSET-SUM, KNAPSACK.
Hamiltonian Cycle

HAM-CYCLE: given an undirected graph $G = (V, E)$, does there exist a simple cycle $\Gamma$ that contains every node in $V$.

YES: vertices and faces of a dodecahedron.

NO: bipartite graph with odd number of nodes.
Claim. DIR-HAM-CYCLE ≤ₚ HAM-CYCLE.

Pf. Given a directed graph $G = (V, E)$, construct an undirected graph $G'$ with $3n$ nodes.

Claim. $G$ has a Hamiltonian cycle iff $G'$ does.

Pf. $\implies$

- Suppose $G$ has a directed Hamiltonian cycle $\Gamma$.
- Then $G'$ has an undirected Hamiltonian cycle (same order).

Pf. $\impliedby$

- Suppose $G'$ has an undirected Hamiltonian cycle $\Gamma'$.
- $\Gamma'$ must visit nodes in $G'$ using one of following two orders:
  - $\ldots$, $B$, $G$, $R$, $B$, $G$, $R$, $B$, $G$, $R$, $B$, $\ldots$
- Blue nodes in $\Gamma'$ make up directed Hamiltonian cycle $\Gamma$ in $G$, or reverse of one. •
3-SAT Reduces to Directed Hamiltonian Cycle

**HAM-CYCLE and DIR-HAM-CYCLE are in NP.**

**Claim.** \(3\text{-SAT} \leq_p \text{DIR-HAM-CYCLE} \) (hence DIR-HAM-CYCLE and HAM-CYCLE are NP-Complete).

**Pf.** Given an instance \(\Phi\) of 3-SAT, we construct an instance of DIR-HAM-CYCLE that has a Hamiltonian cycle iff \(\Phi\) is satisfiable.

**Construction.** First, create a graph that has \(2^n\) Hamiltonian cycles which correspond in a natural way to \(2^n\) possible truth assignments.
3-SAT Reduces to Directed Hamiltonian Cycle

Construction. Given 3-SAT instance $\Phi$ with $n$ variables $x_i$ and $k$ clauses.

- Construct $G$ to have $2^n$ Hamiltonian cycles.
- Intuition: traverse path $i$ from left to right $\iff$ set variable $x_i = 1$.

3-SAT Reduces to Directed Hamiltonian Cycle

Construction. Given 3-SAT instance $\Phi$ with $n$ variables $x_i$ and $k$ clauses.

- For each clause: add a node and 6 edges.
3-SAT Reduces to Directed Hamiltonian Cycle

Claim. \( \Phi \) is satisfiable iff \( G \) has a Hamiltonian cycle.

Pf. \( \Rightarrow \)
- Suppose 3-SAT instance has satisfying assignment \( x^* \).
- Then, define Hamiltonian cycle in \( G \) as follows:
  - if \( x^*_i = 1 \), traverse row \( i \) from left to right
  - if \( x^*_i = 0 \), traverse row \( i \) from right to left
  - for each clause \( C_j \), there will be at least one row \( i \) in which we are going in "correct" direction to splice node \( C_j \) into tour

Pf. \( \Leftarrow \)
- Suppose \( G \) has a Hamiltonian cycle \( \Gamma \).
- If \( \Gamma \) enters clause node \( C_j \), it must depart on mate edge.
  - thus, nodes immediately before and after \( C_j \) are connected by an edge \( e \) in \( G \)
  - removing \( C_j \) from cycle, and replacing it with edge \( e \) yields Hamiltonian cycle on \( G - \{ C_j \} \)
- Continuing in this way, we are left with Hamiltonian cycle \( \Gamma' \) in \( G - \{ C_1, C_2, \ldots, C_k \} \).
- Set \( x^*_i = 1 \) iff \( \Gamma' \) traverses row \( i \) left to right.
- Since \( \Gamma' \) visits each clause node \( C_j \), at least one of the paths is traversed in "correct" direction, and each clause is satisfied. \( \blacksquare \)
Chapter 8

NP and Computational Intractability

Traveling Salesperson Problem

**TSP.** Given a set of $n$ cities and a pairwise distance function $d(u, v)$, is there a tour of length $\leq D$?

All 13,509 cities in US with a population of at least 500
Reference: http://www.tsp.gatech.edu
Traveling Salesperson Problem

**TSP.** Given a set of $n$ cities and a pairwise distance function $d(u, v)$, is there a tour of length $\leq D$?

Reference: http://www.tsp.gatech.edu

11,849 holes to drill in a programmed logic array
Reference: http://www.tsp.gatech.edu
Traveling Salesperson Problem

**TSP.** Given a set of \( n \) cities and a pairwise distance function \( d(u, v) \), is there a tour of length \( \leq D \)?

**HAM-CYCLE:** given a graph \( G = (V, E) \), does there exists a simple cycle that contains every node in \( V \)?

**Claim.** \( \text{HAM-CYCLE} \leq_p \text{TSP} \).

**Pf.**
- Given instance \( G = (V, E) \) of \( \text{HAM-CYCLE} \), create \( n \) cities with distance function
  \[
  d(u, v) = \begin{cases} 
  1 & \text{if } (u, v) \in E \\
  2 & \text{if } (u, v) \notin E
  \end{cases}
  \]
- TSP instance has tour of length \( \leq n \) iff \( G \) is Hamiltonian.

**Is TSP NP-Complete?**
In-Class Exercise

Determine whether the following statements are T/F:

- Problems in NP cannot be solved in polynomial time.

- If $X$ is in NP, $Y$ is NP-Complete and $X \leq_p Y$, then $X$ is NP-Complete.

- The decision version of the sequence alignment problem (i.e. is there an alignment with score < $a$?) is in NP.

- An NP-Complete problem cannot be solved in polynomial-time.

- Even if $P \neq NP$, there may still be some NP-complete problems that can be solved in polynomial time.

- Recall that SAT is in NP. Thus, there exists a polynomial certifier and certificate that proves a CNF formula $X$ is not satisfiable.
8.6-7 Graph Coloring

Basic genres:
- Packing problems: SET-PACKING, INDEPENDENT SET.
- Covering problems: SET-COVER, VERTEX-COVER.
- Constraint satisfaction problems: SAT, 3-SAT.
- Sequencing problems: HAMILTONIAN-CYCLE, TSP.
- Partitioning problems: 3D-MATCHING, 3-COLOR.
- Numerical problems: SUBSET-SUM, KNAPSACK.
3-Colorability

**3-COLOR**: Given an undirected graph $G$ does there exists a way to color the nodes red, green, and blue so that no adjacent nodes have the same color?

![Graph with 3-coloring]

Register Allocation

**Register allocation.** Assign program variables to machine register so that no more than $k$ registers are used and no two program variables that are needed at the same time are assigned to the same register.

**Interference graph.** Nodes are program variables names, edge between $u$ and $v$ if there exists an operation where both $u$ and $v$ are "live" at the same time.

**Observation.** [Chaitin 1982] Can solve register allocation problem iff interference graph is $k$-colorable.

**Fact.** $3\text{-COLOR} \leq_P k\text{-REGISTER-ALLOCATION}$ for any constant $k \geq 3.$
3-Colorability

3-COLOR is in NP.

Claim. 3-SAT \leq_p 3-COLOR (and hence 3-COLOR is NP-Complete).

Pf. Given 3-SAT instance \( \Phi \), we construct an instance of 3-COLOR that is 3-colorable iff \( \Phi \) is satisfiable.

Construction.
i. Make a node for each literal and its negation, and connect the two.
ii. Create 3 new nodes T, F, B: connect them in a triangle, and connect each literal to B.
iii. For each clause, add gadget of 6 nodes and 13 edges.

Claim. Graph is 3-colorable iff \( \Phi \) is satisfiable.

Pf. \( \Rightarrow \) Suppose graph is 3-colorable.

- Consider assignment that sets all T literals to true.
- (i) ensures a literal and its negation are not the same.
- (ii) ensures each literal is T or F.
- (iii) ...
3-Colorability

**Claim.** Graph is 3-colorable iff $\Phi$ is satisfiable.

**Pf.** $\Rightarrow$ Suppose graph is 3-colorable.

- Consider assignment that sets all T literals to true.
- (i) ensures a literal and its negation are not the same.
- (ii) ensures each literal is T or F.
- (iii) ensures at least one literal in each clause is T.

$$C_i = x_1 \lor \overline{x_2} \lor x_3$$
3-Colorability

Claim. Graph is 3-colorable iff $\Phi$ is satisfiable.

Pf. $\Leftarrow$ Suppose 3-SAT formula $\Phi$ is satisfiable.

- Color all true literals T.
- Color node below green node F, and node below that B.
- Color remaining middle row nodes B.
- Color remaining bottom nodes T or F as forced.

\[ C_i = x_1 \lor \overline{x_2} \lor x_3 \]

true T

false F

 literal set to true in 3-SAT as

Chapter 8

NP and Computational Intractability

Slides by Kevin Wayne.
Copyright © 2005 Pearson-Addison Wesley.
All rights reserved.
8.8 Numerical Problems

Basic genres.
- Packing problems: SET-PACKING, INDEPENDENT SET.
- Covering problems: SET-COVER, VERTEX-COVER.
- Constraint satisfaction problems: SAT, 3-SAT.
- Sequencing problems: HAMILTONIAN-CYCLE, TSP.
- Partitioning problems: 3-COLOR, 3D-MATCHING.
- Numerical problems: SUBSET-SUM, KNAPSACK.

Subset Sum

**SUBSET-SUM.** Given natural numbers \( w_1, \ldots, w_n \) and an integer \( W \), is there a subset that adds up to exactly \( W \)?

**Ex:** \( \{ 1, 4, 16, 64, 256, 1040, 1041, 1093, 1284, 1344 \} \), \( W = 3754 \).

Yes. \( 1 + 16 + 64 + 256 + 1040 + 1093 + 1284 = 3754 \).

**SUBSET-SUM (and KNAPSACK) are in NP.**

**Claim.** \( 3\text{-SAT} \leq P \text{ SUBSET-SUM} \) (and hence SUBSET-SUM and KNAPSACK are NP-Complete).

**Pf.** Given an instance \( \Phi \) of 3-SAT, we construct an instance of SUBSET-SUM that has solution iff \( \Phi \) is satisfiable.
### Subset Sum

**Construction.** Given 3-SAT instance \( \Phi \) with \( n \) variables and \( k \) clauses, form \( 2n + 2k \) decimal integers, each of \( n+k \) digits, as illustrated below.

**Claim.** \( \Phi \) is satisfiable iff there exists a subset that sums to \( W \).

**Pf.** No carries possible.

\[
\begin{array}{c|c|c|c|c|c}
\text{C}_1 & \text{C}_2 & \text{C}_3 & x & y & z \\
\hline
100 & 0,10 & 0 & 1 & 0 & 1 & 0 & 10,010 \\
10 & 10 & 1 & 0 & 1 & 1 & 0,101 \\
10 & 10 & 0 & 1 & 1 & 1 & 10,011 \\
1,110 & 1 & 1 & 1 & 0 & 1 & 1,110 \\
1,001 & 1 & 0 & 1 & 0 & 0 & 1,001 \\
100 & 1 & 0 & 0 & 1 & 0 & 0,001 \\
200 & 0 & 0 & 2 & 0 & 0 & 200 \\
10 & 0 & 0 & 0 & 1 & 0 & 0,010 \\
20 & 0 & 0 & 0 & 2 & 0 & 0,020 \\
1 & 0 & 0 & 0 & 0 & 1 & 0,011 \\
2 & 0 & 0 & 0 & 0 & 2 & 0,021 \\
111,444 & 1 & 1 & 1 & 4 & 4 & 111,444
\end{array}
\]

- Dummies to get clause columns to sum to 4.
- Wolfram.com 387.html

### My Hobby:

**Embedding NP-complete problems in restaurant orders**

- **Appetizers**
  - Mixed Fruit: 2.15
  - French Fries: 2.75
  - Side Salad: 3.35
  - Hot Wings: 3.55
  - Mozzarella Sticks: 4.20
  - Sampler Plate: 5.80

- **Sandwiches**
  - Barbeque: 6.50

- **EXACTLY?** Um...

- **Might help you out?**

- **Listen, I have six other tables to get to.**

- **As fast as possible.**