

CSEP 521
Applied Algorithms
Spring 2005

Traveling Salesman Problem
NP-Completeness

Reading

- Chapter 34
- Chapter 35

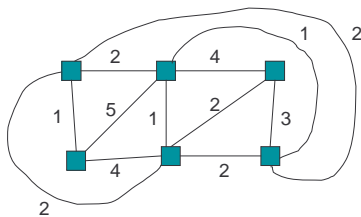
Outline for the Evening

- Traveling Salesman Problem
 - Approximation algorithms
 - Local search algorithms
- P and NP
- Reducibility and NP-Completeness
- Clique, Colorability, and other NP-complete problems
- Coping with NP-completeness

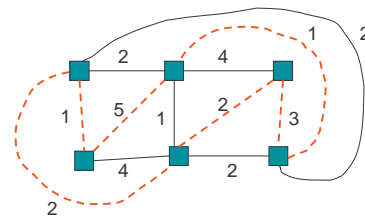
Traveling Salesman Problem

- Input: Undirected Graph $G = (V, E)$ and a cost function C from E to the reals. $C(e)$ is the cost of edge e .
- Output: A cycle that visits each vertex exactly once and is minimum total cost.

Example



Example



$$\text{Cost} = 1 + 5 + 1 + 3 + 2 + 2 = 14$$

Variations

- Hamiltonian Cycle
 - Is there a cycle that visits each vertex exactly once
 - Ignores costs
- Triangle inequality constraint
 - $C(u,v) \leq C(u,x) + C(x,v)$
- Euclidean Traveling Salesman
 - Vertices are points on the plane and the cost is the Euclidean distance between them
 - Implies triangle inequality

Lecture 2 - Traveling Salesman,
NP-Completeness

7

Applications

- Telescope planning
- Route planning
 - coin pickup
 - mail delivery
 - book order pickup in the Amazon warehouse
- Circuit board drilling

Lecture 2 - Traveling Salesman,
NP-Completeness

8

Why Traveling Salesman?

- Old well-studied problem
- Example of an NP-hard problem
 - These problems are very hard to solve exactly
 - No polynomial time algorithms known to exist
- Interesting and effective approximation algorithms
 - Good practical algorithms
 - Simple algorithms with provable approximation bounds

Lecture 2 - Traveling Salesman,
NP-Completeness

9

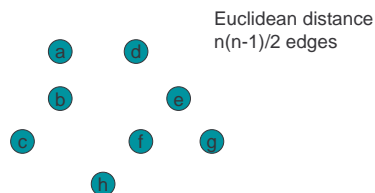
Approximation Alg. vs. Heuristic

- Approximation Algorithm
 - There is a provable guarantee of how close the algorithm's result is to the optimal solution.
- Heuristic
 - The algorithm finds a solution but there is no guarantee how good the solution is.
 - Heuristics often outperform provable approximation algorithms.

Lecture 2 - Traveling Salesman,
NP-Completeness

10

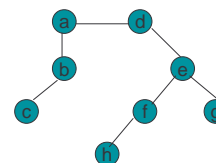
A Simple Approximation Algorithm



Lecture 2 - Traveling Salesman,
NP-Completeness

11

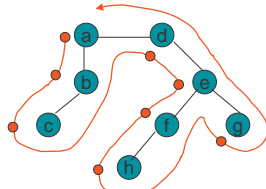
1. Find a Minimum Spanning Tree



Lecture 2 - Traveling Salesman,
NP-Completeness

12

2. Depth-First Search of Tree

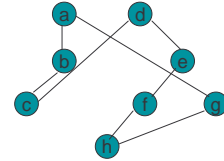


Marking Order = a, b, c, d, e, f, h, g

Lecture 2 - Traveling Salesman,
NP-Completeness

13

3. Connect Vertices in Marking Order



Marking Order = a, b, c, d, e, f, h, g

Lecture 2 - Traveling Salesman,
NP-Completeness

14

Evaluation

- Time and Storage
 - Time $O(n^2 \log n)$ with Kruskal's Algorithm
 - Storage $O(n^2)$
- Quality of Solution H
 - $C(H) \leq 2 C(H^*)$ where H^* is an optimal tour
 - This is a “2-approximation algorithm”
- Same approximation bound applies to case of triangle inequality

Lecture 2 - Traveling Salesman,
NP-Completeness

15

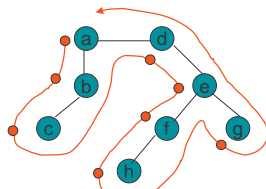
Proof of Approximation Bound

- Setup
 - T minimum spanning tree
 - W the depth-first walk of T
 - H the tour computed by the algorithms
 - H^* an optimal tour

Lecture 2 - Traveling Salesman,
NP-Completeness

16

Depth-First Walk



$C(W) = 2 C(T)$
 $C(H) \leq C(W)$
 triangle inequality

Depth-first walk = a, b, c, b, a, d, e, f, h, f, e, g, e, d, a
 Marking order = a, b, c, d, e, f, h, g

Lecture 2 - Traveling Salesman,
NP-Completeness

17

Proof of Approximation Bound

1. $C(W) = 2 C(T)$
2. $C(H) \leq C(W)$, triangle inequality
3. $C(H) \leq 2 C(T)$, last two lines
4. $C(T) \leq C(H^*)$, minus an edge H^* is a spanning tree
5. $C(H) \leq 2 C(H^*)$, last two lines

Lecture 2 - Traveling Salesman,
NP-Completeness

18

Solving TSP Exactly

- Branch-and-Bound
 - $n < 25?$
- Linear Programming
 - $n < 100$
- Cutting Plane Methods for Euclidian case
 - $n < 15,000$ with "concord"
 - see <http://www.math.princeton.edu/tsp/>

Lecture 2 - Traveling Salesman,
NP-Completeness

19

Solving TSP Approximately

- 3/2 – approximation algorithm of Christofedes
- Polynomial approximation scheme for Euclidian TSP by Aurora (1998), Mitchell (1999)
 - To get within $(1+\epsilon)$ of optimal can be done in time polynomial in $1/\epsilon$ and n .
 - These are not practical

Lecture 2 - Traveling Salesman,
NP-Completeness

20

Solving TSP Approximately, Practically

- Local Search
 - Lin-Kernighan method
- Simulated Annealing
- Genetic Algorithms
- Neural Networks

Lecture 2 - Traveling Salesman,
NP-Completeness

21

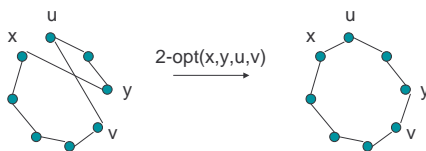
Local Search Algorithms

- Start with an initial solution that is usually easy to find, but is not necessarily good.
- Repeatedly modify the current solution to a better nearby one. Until no nearby one is better.

Lecture 2 - Traveling Salesman,
NP-Completeness

22

2-Opt Neighborhood



Lecture 2 - Traveling Salesman,
NP-Completeness

23

2-opt Algorithm

Lin-Kernighan (1973)

```

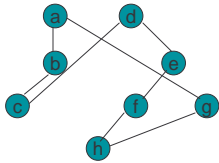
Find an initial tour T
1. For every pair of distinct edges (x,y), (u,v) in T
   if  $C(x,u) + C(y,v) < C(x,y) + C(u,v)$  then
      $T := T - \{(x,y), (u,v)\} \cup \{(x,u), (y,v)\}$ 
     exit for loop and go to 1
Return T
    
```

Lecture 2 - Traveling Salesman,
NP-Completeness

24

Example of LK

Euclidian case

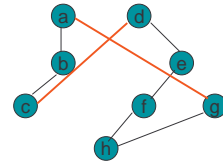


Lecture 2 - Traveling Salesman,
NP-Completeness

25

Example of LK

Euclidian case

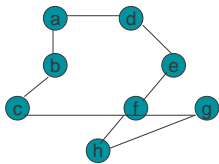


Lecture 2 - Traveling Salesman,
NP-Completeness

26

Example of LK

Euclidian case

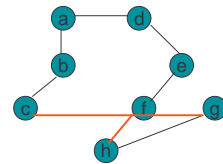


Lecture 2 - Traveling Salesman,
NP-Completeness

27

Example of LK

Euclidian case

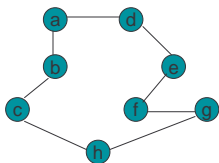


Lecture 2 - Traveling Salesman,
NP-Completeness

28

Example of LK

Euclidian case

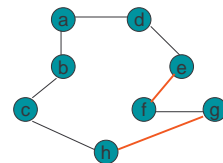


Lecture 2 - Traveling Salesman,
NP-Completeness

29

Example of LK

Euclidian case

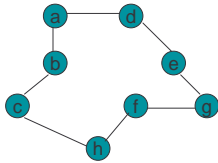


Lecture 2 - Traveling Salesman,
NP-Completeness

30

Example of LK

Euclidian case



Lecture 2 - Traveling Salesman,
NP-Completeness

31

Lin-Kernighan

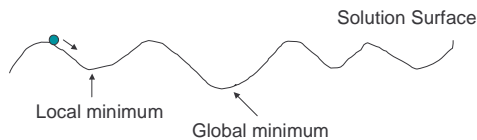
- Empirical $O(n^{2.2})$ time
- Finds optimal in most examples < 100 points
- Excellent Implementations
 - Can easily handle hundreds of thousands of points

Lecture 2 - Traveling Salesman,
NP-Completeness

32

Local Minimum Problem

- Local search can lead to a local minimum in the solution space, not necessarily a global minimum.



Lecture 2 - Traveling Salesman,
NP-Completeness

33

NP-Completeness Theory

- Explains why some problems are hard and probably not solvable in polynomial time.
- Invented by Cook in 1971.
- Popularized in an important paper by Karp in 1972.
- Standardized by Garey and Johnson in 1979 in "Computers and Intractability: A Guide to the Theory of NP-Completeness".

Lecture 2 - Traveling Salesman,
NP-Completeness

34

P

- Complexity theory is the study of the time and storage needed to solve problems.
 - Sorting requires $\Theta(n \log n)$ time
 - Minimum spanning tree can be solved in $O(m \log m)$ time
 - Connected components can be solved in $O(m)$ time.
- P is the class of problems that can be solved in polynomial time.
 - $O(n), O(n^2), O(n^3), \dots$ time

Lecture 2 - Traveling Salesman,
NP-Completeness

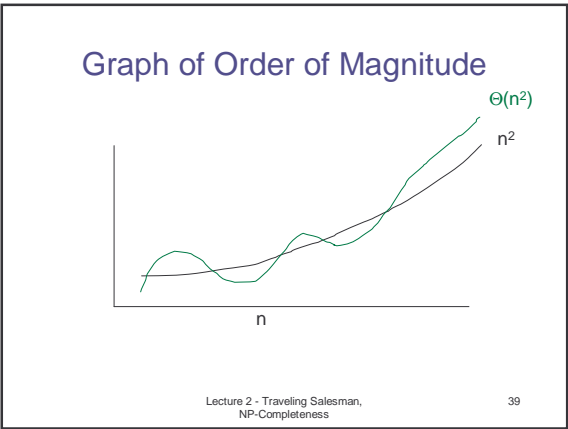
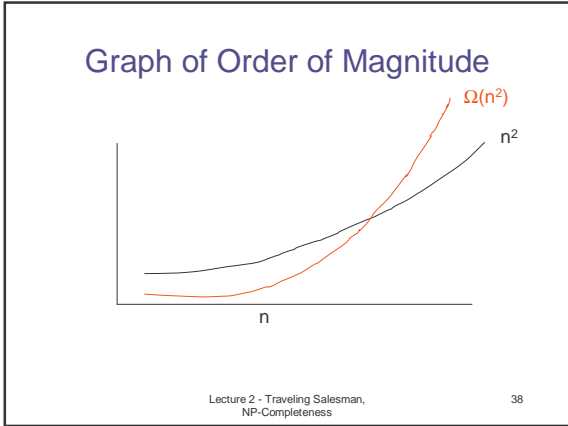
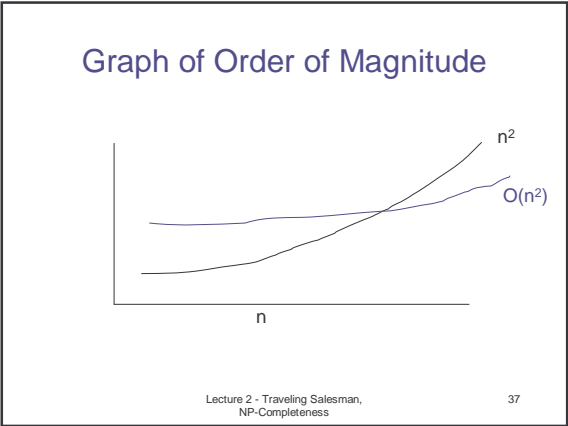
35

Order Notation

- $f(n) = O(g(n))$ means $f(n) \leq c g(n)$ for some c .
 - $1,000,000 n^2 + 2n = O(n^2)$
 - $n \log n = O(n^3)$
- $f(n) = \Omega(g(n))$ means $f(n) \geq c g(n)$ for some $c > 0$.
 - $.0000001 n^2 + 2n = \Omega(n^2)$
 - $1,000 n^2 = \Omega(n)$
- $f(n) = \Theta(g(n))$ means $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$
 - $a_k n^k + a_{k-1} n^{k-1} + \dots = \Theta(n^k)$ if $a_k > 0$

Lecture 2 - Traveling Salesman,
NP-Completeness

36



Worst Case Asymptotic Analysis

- Given problem find the best $t(n)$ such that there is an algorithm solving the problem that runs in time $O(t(n))$ on all inputs of size n .
 - $t(n)$ is an **asymptotic upper bound**
- Given a problem find the best $t'(n)$ such that every algorithm solving the problem runs in time $\Omega(t'(n))$ on some input of length n .
 - $t'(n)$ is an **asymptotic lower bound**

Lecture 2 - Traveling Salesman, NP-Completeness 40

Bane of Worst Case Asymptotic Analysis

- Worst case**
 - A bad asymptotic algorithm in the worst case might do well on the common case.
- Asymptotic**
 - A good asymptotic algorithm might perform poorly on inputs of reasonable size.

Lecture 2 - Traveling Salesman, NP-Completeness 41

NP

- NP stands for nondeterministic polynomial time.
- We consider the class of decision problems (yes/no problems).
- A nondeterministic algorithm is one that can make "guesses".
- A decision problem is in NP if it can be solved by a nondeterministic algorithm that runs in polynomial time.
- Some problems in NP seem very hard to solve.

Lecture 2 - Traveling Salesman, NP-Completeness 42

Examples of Decision Problems in NP

- Decision TSP
 - Input: Graph $G = (V, E)$ with costs on the edges and a budget B
 - Output: Determine if there is a tour visiting each vertex exactly once of total cost $\leq B$.
 - Algorithm: Guess a tour and check its cost is under budget.
- Graph Coloring
 - input: Graph $G = (V, E)$ and a number k .
 - output: Determine if all vertices can be colored with k colors such that no two adjacent vertices have the same color.
 - Algorithm: Guess a coloring and then check it.

Lecture 2 - Traveling Salesman,
NP-Completeness

43

CNF-SAT

- Input: A Boolean formula F in conjunctive normal form.

$$(x \vee y \vee z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$$
- Output: Determine if F is satisfiable, that is, there is some assignment to the variables that makes the formula F true.

$$x = 1, y = 0, z = 1$$

$$(1 \vee 0 \vee 1) \wedge (\neg 1 \vee 0 \vee 1) \wedge (\neg 1 \vee \neg 0 \vee \neg 1)$$
- Algorithm: Guess an assignment and check it.

Lecture 2 - Traveling Salesman,
NP-Completeness

44

Subset Sum

- Input: Integers a_1, a_2, \dots, a_n, b
- Output: Determine if there is subset $X \subseteq \{1, 2, \dots, n\}$ with the property $\sum_{i \in X} a_i = b$
- Algorithm: Guess the subset X and check the sum adds up to b .

Lecture 2 - Traveling Salesman,
NP-Completeness

45

Decision Problems Reporting Problems Optimization Problems

- Example 1: Subset sum
 - Decision Problem: Determine if a subset sum exists.
 - Reporting Problem: If a subset sum exists, then report one.
 - Optimization Problem: Find a subset whose sum is as close as possible to b , without going over b .

Lecture 2 - Traveling Salesman,
NP-Completeness

46

Decision Problems Reporting Problems Optimization Problems

- Example 2. Traveling Salesman
 - Optimization problem – Find a tour that minimizes cost.
 - Decision problem – Determine if a tour exists that comes under a specified budget.
 - Reporting problem - If a tour exist that comes under a specified budget, find it.

Lecture 2 - Traveling Salesman,
NP-Completeness

47

Polynomial Time Equivalence of Decision, Reporting, Optimization

- If any one of Decision, Reporting, or Optimization can be solved in polynomial time then so can the others.
- Decision is easily reducible to Optimization
 - Subset sum
 - Traveling salesman

Lecture 2 - Traveling Salesman,
NP-Completeness

48

Reporting Reduces to Decision

- Subset sum:
 - Let subset-sum(A, b) return true if some subset of A adds up to b . Otherwise it returns false.

```

Precondition: subset-sum ( $\{a_1, \dots, a_n\}, b$ ) is true
Report ( $\{a_1, \dots, a_n\}, b$ )
 $X :=$  the empty set;
for  $i = 1$  to  $n$  do
    if subset-sum( $\{a_{i+1}, \dots, a_n\}, b - a_i$ ) then
        add  $i$  to  $X$ ;
         $b := b - a_i$ ;
    
```

Example

3, 5, 2, 7, 4, 2, $b = 11$

5, 2, 7, 4, 2, $b = 11 - 3 \rightarrow$ yes, $X = \{3\}$, $b = 8$

2, 7, 4, 2, $b = 8 - 5 \rightarrow$ no

7, 4, 2, $b = 8 - 2 \rightarrow$ yes, $X = \{3, 2\}$, $b = 6$

4, 2, $b = 6 - 7 \rightarrow$ no

2, $b = 6 - 4 \rightarrow$ yes, $X = \{3, 2, 4\}$, $b = 2$

$b = 2 - 2 \rightarrow$ yes, $X = \{3, 2, 4, 2\}$

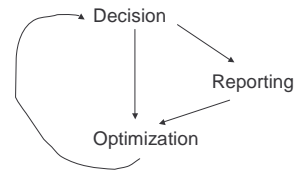
Optimization Reduces to Decision

- Traveling Salesman
 - $TS(G, B)$ which returns true if and only if G has a tour of length $\leq B$. Assume costs are positive integers.
1. Find the minimum cost of a tour by binary search
 2. Find the tour itself (reporting).

```

Find minimum cost of a tour
 $L := 0$ ;
 $U :=$  sum of all costs of edges;
while  $L + 1 < U$  do
     $B = (L + U) / 2$ ;
    if  $TS(G, B)$  then  $U := B$  else  $L := B$ ;
return  $U$ 
    
```

The Relationship



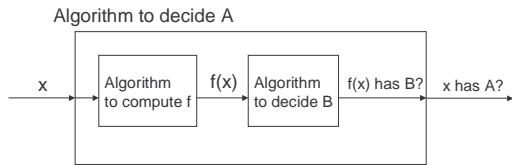
Exercise

1. Assume the decision algorithm subset-sum(A, b) is provided. Solve the optimization problem for subset sum.
2. Assume the decision problem $TS(G, B)$ is given. Solve the reporting problem for traveling salesman.

Polynomial Time Reducibility

- Informal idea: A decision problem A is polynomial time reducible to a decision problem B if a polynomial time algorithm for B can be used to construct a polynomial time algorithm for A .
- Formally: A is polynomial time reducible to B if there is a function f computable in polynomial time such that for all x :
 - x has A if and only if $f(x)$ has B
- If A polynomial time reducible to B and B solvable in polynomial time then so is A .

Block Diagram to Decide A from B



Lecture 2 - Traveling Salesman,
NP-Completeness

55

Transitivity of Polynomial Time Reduction

- Define: $A \leq_p B$ to mean that A is polynomial time reducible to B .
- Transitivity: $A \leq_p B$ and $B \leq_p C$ implies $A \leq_p C$
- Example:
 - Every problem in NP is known to be polynomial time reducible to CNF-SAT.
 - SAT is polynomial time reducible to Decision TSP
 - Therefore, every problem in NP is polynomial time reducible to Decision TSP.

Lecture 2 - Traveling Salesman,
NP-Completeness

56

NP-Completeness Definition

- Definition: A decision problem A is NP-complete if
 - A is in NP
 - Every problem in NP is reducible to A in polynomial time.
- NP-complete problems seem to require exponential time, but there is no proof to date.

Lecture 2 - Traveling Salesman,
NP-Completeness

57

Cook's Theorem

- CNF-satisfiability is NP-complete
 - Cook 1971, Levin 1973

Proof formalizes the notion of a nondeterministic algorithm as a nondeterministic Turing machine. It can be shown that a CNF-formula F can be produced in polynomial time that describes the operation of the nondeterministic Turing machine. The Turing machine halts in a "yes" state if and only if the formula F is satisfiable.

Lecture 2 - Traveling Salesman,
NP-Completeness

58

NP-Hardness

- Definition: A problem A is NP-hard if an NP-complete problem can be solved using A as an "oracle".
 - Decision TSP is NP-complete
 - TSP is NP-hard
- Oracle is like a constant time function call.

Lecture 2 - Traveling Salesman,
NP-Completeness

59

P vs NP

- Every problem in P is also in NP

$$P \subseteq NP$$

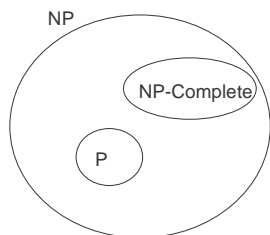
- Famous Unsolved Open Question:

$$P = NP?$$

Lecture 2 - Traveling Salesman,
NP-Completeness

60

Probable Picture



Lecture 2 - Traveling Salesman,
NP-Completeness

61

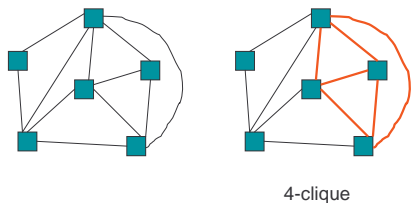
Clique Decision Problem

- Input: Undirected Graph $G = (V, E)$ and a number k .
- Output: Determine if G has a k -clique, that is, there is a set of vertices U of size k such that for each pair of vertices in U there is an edge in E between them.

Lecture 2 - Traveling Salesman,
NP-Completeness

62

Clique Example



4-clique

Lecture 2 - Traveling Salesman,
NP-Completeness

63

Clique is NP-Complete

- Clique is in NP
 - Nondeterministic algorithm: guess k vertices then check that there is an edge between each pair of them.
- Clique is NP-hard
 - We reduce CNF-satisfiability to Clique in polynomial time
 - Given a CNF formula F we need to construct a graph G and a number k with the property that F is satisfiable if and only if G has a k -clique. The construction must be efficient, polynomial time.

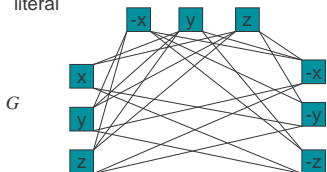
Lecture 2 - Traveling Salesman,
NP-Completeness

64

Construction by Example

$$F = (x \vee y \vee z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$$

literal clause



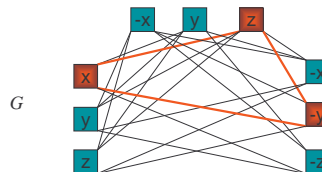
Lecture 2 - Traveling Salesman,
NP-Completeness

65

Construction by Example

$$F = (x \vee y \vee z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$$

$x = 1, y = 0, z = 1$



Lecture 2 - Traveling Salesman,
NP-Completeness

66

General Construction

$$F = \bigcap_{i=1}^k \bigcup_{j=1}^{m_i} a_{ij} \quad \text{where } a_{ij} \in \{x_1, \neg x_1, \dots, x_n, \neg x_n\}$$

↑
literals

$$G = (V, E) \quad \text{where}$$

$$V = \{a_{ij} : 1 \leq i \leq k, 1 \leq j \leq m_i\}$$

$$E = \{\{a_{ij}, a_{i'j'}\} : i \neq i' \text{ and, } a_{ij} \text{ and } a_{i'j'} \text{ are not complementary}\}$$

k is the number of clauses

The Reduction Argument

• We must show

– F satisfiable implies G has a clique of size k .

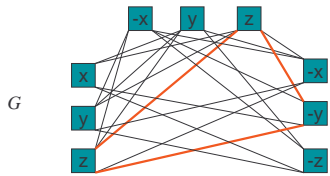
- Given a satisfying assignment for F , for each clause pick a literal that is satisfied. Those literals in the graph G form a k -clique.

– G has a clique of size k implies F is satisfiable.

- Given a k -clique in G , assign each literal in the clique to be 1. This yields a satisfying assignment to F .

Clique to Assignment

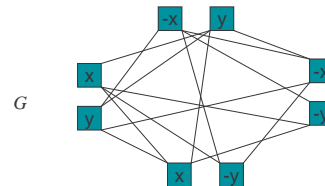
$$F = (x \vee y \vee z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$$



$$y = 0, z = 1$$

Assignment to Clique

$$F = (x \vee y) \wedge (\neg x \vee y) \wedge (\neg x \vee \neg y) \wedge (x \vee \neg y)$$



G has no 4-clique

3-CNF-Satisfiability

- Input: A Boolean formula F with at most 3 literals per clause.
- Output: Determine if F is satisfiable.
- 3-CNF-Satisfiability is NP-complete
 - This is probably the most used NP-complete problem in reduction proofs showing other decision problems are NP-hard.

Reduction by Example

Given $F = (x_1 \vee \neg x_2 \vee x_3 \vee \neg x_4) \wedge F'$

Construct $H = (x_1 \vee z_1) \wedge (\neg x_2 \vee \neg z_1 \vee z_2) \wedge (x_3 \vee \neg z_2 \vee z_3) \wedge (\neg x_4 \vee \neg z_3) \wedge F'$

F is satisfiable if and only if H is satisfiable.

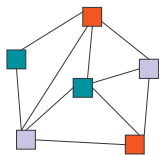
$x_2 = 0$ satisfies the first clause of F .

$z_1 = 1, z_2 = 0, z_3 = 0$ satisfy clauses 1,3, and 4 of H and

$x_2 = 0$ satisfies the clause 2 of H .

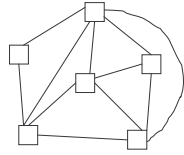
3-Colorability

- Input: Graph $G = (V, E)$.
- Output: Determine if all vertices can be colored with 3 colors such that no two adjacent vertices have the same color.



3-colorable

Lecture 2 - Traveling Salesman,
NP-Completeness



Not 3-colorable

73

3-CNF-Sat \leq_p 3-Color

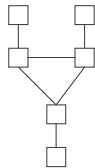
- Given a 3-CNF formula F we have to show how to construct in polynomial time a graph G such that:
 - F is satisfiable implies G is 3-colorable
 - G is 3-colorable implies F is satisfiable

Lecture 2 - Traveling Salesman,
NP-Completeness

74

The Gadget

- This is a classic reduction that uses a “gadget”.
- Assume the outer vertices are colored at most two colors. The gadget is 3-colorable if and only if the outer vertices are not all the same color.

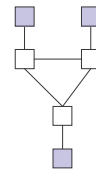


Lecture 2 - Traveling Salesman,
NP-Completeness

75

Properties of the Gadget

- Three colorable if and only if outer vertices not all the same color.



Not 3 colorable



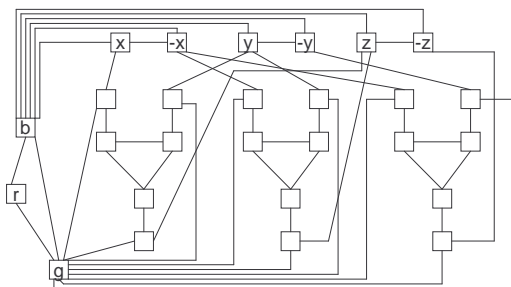
Is 3 colorable

Lecture 2 - Traveling Salesman,
NP-Completeness

76

Reduction by Example

$$F = (x \vee y \vee z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$$



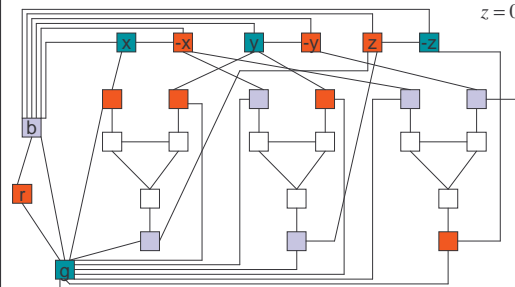
Lecture 2 - Traveling Salesman,
NP-Completeness

77

Satisfaction Example

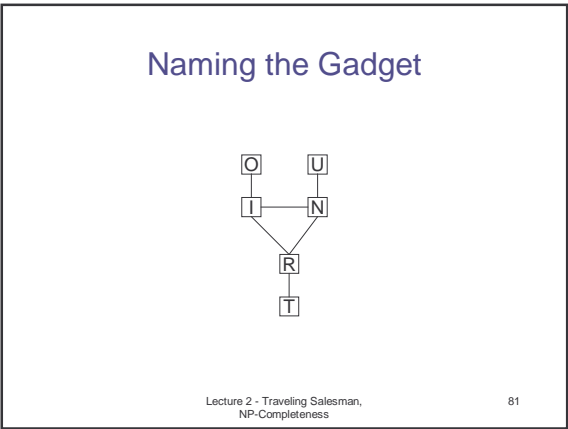
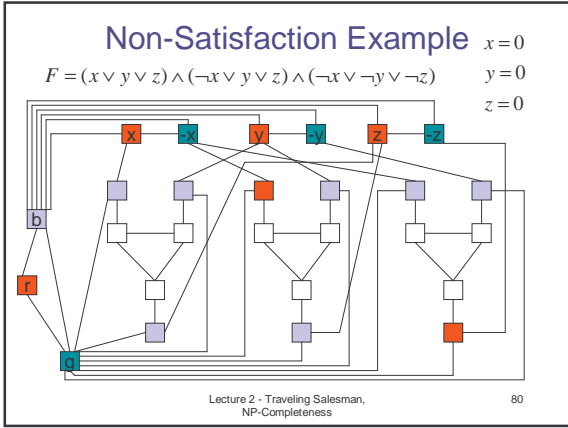
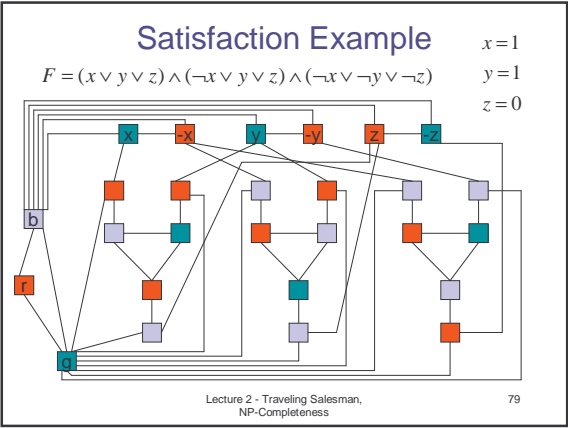
$$F = (x \vee y \vee z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$$

$$\begin{aligned} x &= 1 \\ y &= 1 \\ z &= 0 \end{aligned}$$



Lecture 2 - Traveling Salesman,
NP-Completeness

78



General Construction

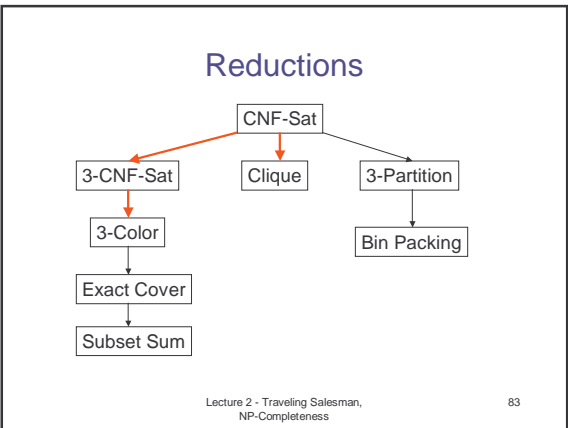
$F = \bigwedge_{i=1}^k (a_{i1} \vee a_{i2} \vee a_{i3})$ where $a_{ij} \in \{x_1, \neg x_1, \dots, x_n, \neg x_n\}$

$G = (V, E)$ where

$V = \{r, g, b\} \cup \{x_1, \neg x_1, \dots, x_n, \neg x_n\} \cup \{O_i, U_i, T_i, I_i, N_i, R_i : 1 \leq i \leq k\}$

$E = \{ \{r, g\}, \{g, b\}, \{b, r\} \}$
 $\cup \{ \{x_i, \neg x_i\}, \dots, \{x_n, \neg x_n\} \}$
 $\cup \{ \{x_i, b\}, \{-x_i, b\}, \dots, \{x_n, b\}, \{-x_n, b\} \}$
 $\cup \{ \{O_i, I_i\}, \{U_i, N_i\}, \{T_i, R_i\}, \{I_i, N_i\}, \{N_i, R_i\}, \{R_i, I_i\} : 1 \leq i \leq k \}$
 $\cup \{ \{a_{i1}, O_i\}, \{a_{i2}, U_i\}, \{a_{i3}, T_i\} : 1 \leq i \leq k \}$
 $\cup \{ \{O_i, g\}, \{U_i, g\}, \{T_i, g\} : 1 \leq i \leq k \}$

Lecture 2 - Traveling Salesman, NP-Completeness 82



Exact Cover

- Input: A set $U = \{u_1, u_2, \dots, u_n\}$ and subsets $S_1, S_2, \dots, S_m \subseteq U$
- Output: Determine if there is set of pairwise disjoint sets that union to U , that is, a set X such that:

$$X \subseteq \{1, 2, \dots, m\}$$

$$i, j \in X \text{ and } i \neq j \text{ implies } S_i \cap S_j = \emptyset$$

$$\bigcup_{i \in X} S_i = U$$

Lecture 2 - Traveling Salesman, NP-Completeness 84

Example of Exact Cover

$$U = \{a, b, c, d, e, f, g, h, i\}$$

$$\{a, c, e\}, \{a, f, g\}, \{b, d\}, \{b, f, h\}, \{e, h, i\}, \{f, h, i\}, \{d, g, i\}$$

Exact Cover

$$\{a, c, e\}, \{b, f, h\}, \{d, g, i\}$$

3-Partition

- Input: A set of numbers $A = \{a_1, a_2, \dots, a_{3m}\}$ and number B with the properties that $B/4 < a_i < B/2$ and

$$\sum_{i=1}^{3m} a_i = mB.$$

- Output: Determine if A can be partitioned into S_1, S_2, \dots, S_m such that for all i

$$\sum_{j \in S_i} a_j = B.$$

Note: each S_i must contain exactly 3 elements.

Example of 3-Partition

- $A = \{26, 29, 33, 33, 33, 34, 35, 36, 41\}$
- $B = 100, m = 3$
- 3-Partition
 - 26, 33, 41
 - 29, 36, 35
 - 33, 33, 34

Bin Packing

- Input: A set of numbers $A = \{a_1, a_2, \dots, a_m\}$ and numbers B (capacity) and K (number of bins).
- Output: Determine if A can be partitioned into S_1, S_2, \dots, S_K such that for all i

$$\sum_{j \in S_i} a_j \leq B.$$

Bin Packing Example

- $A = \{2, 2, 3, 3, 3, 4, 4, 4, 5, 5, 5\}$
- $B = 10, K = 4$
- Bin Packing
 - 3, 3, 4
 - 2, 3, 5 Perfect fit!
 - 5, 5
 - 2, 4, 4

Exercise – Argue NP-Completeness

1. Independent Set

- Input: Undirected graph $G = (V, E)$ and a number k .
- Output: Determine if there is an independent set of size k . X , contained in V , is independent if for all i, j in X there is no edge in G from i to j .

2. Equal Subset-Sum

- Input: $\{a_1, a_2, \dots, a_n\}$ positive integers
- Output: Determine if there is a set I such that

$$\sum_{i \in I} a_i = \sum_{j \notin I} a_j$$

Coping with NP-completeness

- You have encountered a Hard Problem
- Maybe it is NP-hard
 - Books
 - Garey and Johnson
 - Websites
 - <http://www.nada.kth.se/~viggo/problemist/compendium.htm>
 - Research papers
 - Maybe you'll have to do your own reduction
- Can't determine NP-hardness, then it is probably hard in some way.
- Modify the problem to be more tractable

Lecture 2 - Traveling Salesman,
NP-Completeness

91

Boundary Between P and NP

- Satisfiability
 - 2-CNF-SAT is in P
 - 3-CNF-SAT is NP-complete
- Coloring
 - 2-COLOR is in P
 - 3-COLOR is NP-complete
- Planar Colorability
 - Planar graphs are always 4-colorable
 - 3-PLANAR-COLOR is NP-complete

Lecture 2 - Traveling Salesman,
NP-Completeness

92

Boundary Continued

- Independent Set
 - Maximum independent set is NP-hard
 - Maximal independent set is in P
- Cutting a graph
 - Maximum cut in a graph is NP-hard
 - Minimum cut in a graph is in P (equivalent to Max Flow)
- Spanning Tree
 - Minimum spanning tree is in P
 - Degree constrained spanning tree is NP-hard
 - Bounded diameter spanning tree is NP-hard

Lecture 2 - Traveling Salesman,
NP-Completeness

93

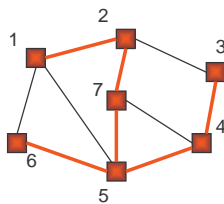
Load Balanced Spanning Tree

- Input: An undirected graph $G = (V, E)$.
- Output: A number k and a spanning tree (V, T) of degree k . Furthermore, there is no spanning tree of degree $< k$.

Lecture 2 - Traveling Salesman,
NP-Completeness

94

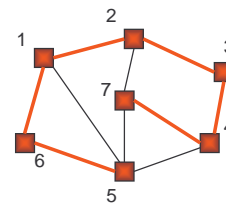
Spanning Tree of Degree 3



Lecture 2 - Traveling Salesman,
NP-Completeness

95

Spanning Tree of Degree 2



Lecture 2 - Traveling Salesman,
NP-Completeness

96

LBST Decision Problem

- Input: An undirected graph $G = (V, E)$ and number k .
- Output: Determine if G has a spanning tree of degree k .

Lecture 2 - Traveling Salesman,
NP-Completeness

97

Hamiltonian Path Decision Problem

- Input: Undirected Graph $G = (V, E)$.
- Output: Determine if there is a path in G that visits each node exactly once.
- Hamiltonian Path is known to be NP-complete

Lecture 2 - Traveling Salesman,
NP-Completeness

98

Hamiltonian Path is Polynomial time Reducible to Spanning Tree of Degree 2

- If there an algorithm to quickly determine if a graph has a spanning tree of degree 2 then there is an algorithm to quickly solve the Hamiltonian path problem.
 - A spanning tree of degree 2 is a Hamiltonian path!
 - These problems are essentially the same problem.

Lecture 2 - Traveling Salesman,
NP-Completeness

99

Lessons When Coping

- Lesson 1. Any problem that is in NP may be NP-complete.
- Lesson 2. Any problem in NP may be in P.
- Lesson 3. You may not be able to determine either
 - factoring is open
 - graph isomorphism is open

Lecture 2 - Traveling Salesman,
NP-Completeness

100