Learning Goals

- Define Hamiltonian Cycles and Paths
- State the HAMILTONIAN CYCLE/PATH problem
- Understand the reduction from HAMILTONIAN CYCLE to HAMILTONIAN PATH
- Understand the reduction from 3-SAT to HAMILTONIAN CYCLE
- State the TRAVELING SALESMAN problem
- Understand the reduction from HAMILTONIAN PATH to TRAVELING SALESMAN
- Definition of Eulerian paths and cycles
- Criterion for existence of Eulerian path/cycle

Classical NP-complete problems

- Richard Karp (1972): Many combinatorial problems commonly encountered are NP-complete!

Classical NP-complete problems

- Richard Karp (1972): Many combinatorial problems commonly encountered are NP-complete!
- Since then, new problems are shown to be NP complete almost every week (if not every day).

Classical NP-complete problems

- Richard Karp (1972): Many combinatorial problems commonly encountered are NP-complete!
- Since then, new problems are shown to be NP complete almost every week (if not every day).
- Different classes of NP-complete problems enable us to recognize other NP-complete problems faster.
- Reductions to these problems are exemplary for coming up with reductions.

Hamiltonian Cycles

Definition

In a directed graph, a *Hamiltonian path* is a path that visits each vertex exactly once. A *Hamiltonian cycle* is a cycle that visits each vertex exactly once.

Hamiltonian Cycles

Definition

In a directed graph, a *Hamiltonian path* is a path that visits each vertex exactly once. A *Hamiltonian cycle* is a cycle that visits each vertex exactly once.

Definition

In the HAMILTONIAN CYCLE problem, we are given a directed graph and must decide whether there exists a Hamiltonian cycle. In the HAMILTONIAN PATH problem, we are given a directed graph and must decide whether there exists a Hamiltonian path.

Theorem

HAMILTONIAN CYCLE is NP*-complete.*

Theorem

HAMILTONIAN CYCLE is NP*-complete.*

Proof.

HAMILTONIAN CYCLE is in NP. Certificate for a YES instance: a Hamiltonian cycle. Verifier: verify it is a cycle that visits every node exactly once.

Theorem

HAMILTONIAN CYCLE is NP*-complete.*

Proof.

HAMILTONIAN CYCLE is in NP. Certificate for a YES instance: a Hamiltonian cycle. Verifier: verify it is a cycle that visits every node exactly once.

We show 3-SAT $\leq_P$ HAMILTONIAN CYCLE.

"Gadget": a fragment of a problem that encodes a fragment from another problem.

Theorem

HAMILTONIAN CYCLE is NP*-complete.*

Proof.

HAMILTONIAN CYCLE is in NP. Certificate for a YES instance: a Hamiltonian cycle. Verifier: verify it is a cycle that visits every node exactly once.

We show 3-SAT $\leq_P$ HAMILTONIAN CYCLE.

"Gadget": a fragment of a problem that encodes a fragment from another problem.

Example: the 3-cycles in the reduction from 3-SAT to INDEPENDENT SET represent clauses.

Theorem

HAMILTONIAN CYCLE is NP-*complete.*

Proof.

HAMILTONIAN CYCLE is in NP. Certificate for a YES instance: a Hamiltonian cycle. Verifier: verify it is a cycle that visits every node exactly once.

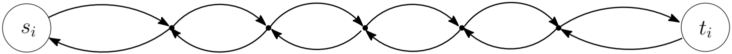
We show 3-SAT $\leq_P$ HAMILTONIAN CYCLE.

"Gadget": a fragment of a problem that encodes a fragment from another problem.

Example: the 3-cycles in the reduction from 3-SAT to INDEPENDENT SET represent clauses.

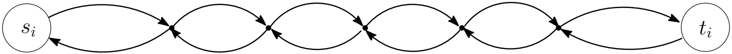
In HAMILTONIAN CYCLE, what is a gadget to represent the TRUE or FALSE assignment to a variable in 3-SAT?

Gadget representing variables

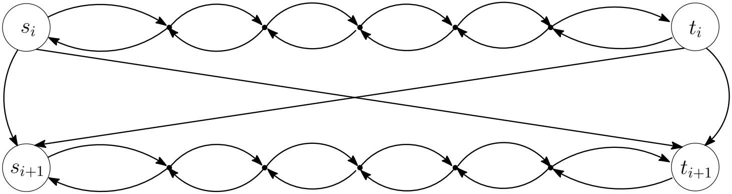


A gadget for a variable. Starting from s_i there is one way to traverse all nodes and arrive at t_i (TRUE); starting from t_i, there is one way to traverse all nodes and arrive at s_i (FALSE).

Gadget representing variables



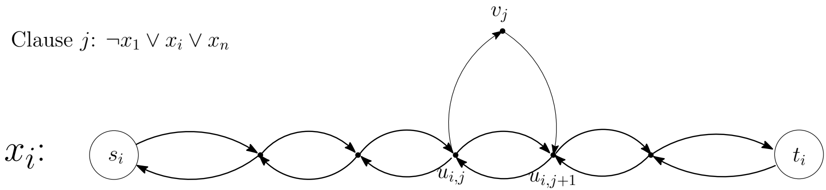
A gadget for a variable. Starting from s_i there is one way to traverse all nodes and arrive at t_i (TRUE); starting from t_i, there is one way to traverse all nodes and arrive at s_i (FALSE).



Connection of two variable gadgets.

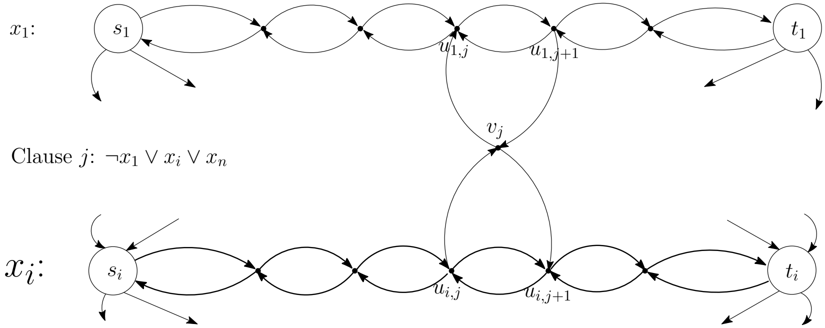
Adding Clauses

Clause j: $\neg x_1 \vee x_i \vee x_n$



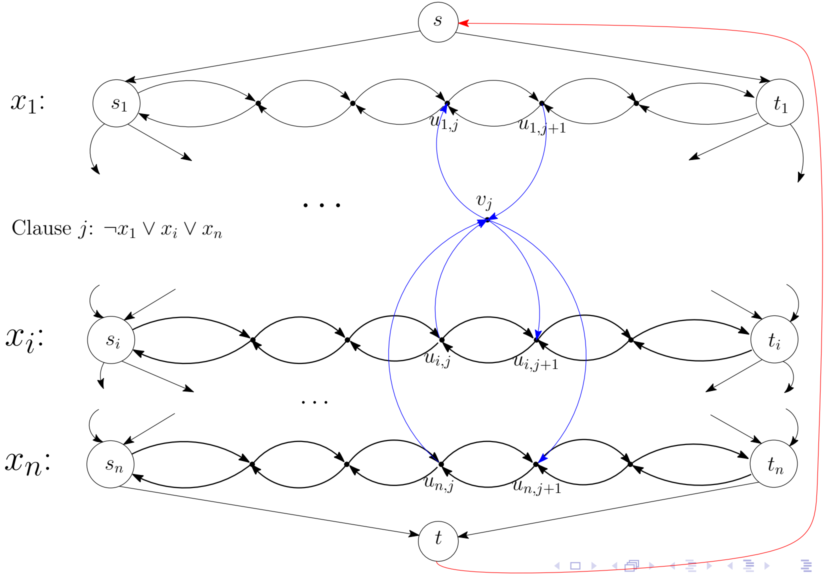
Adding a positive literal to a clause.

Adding Clauses



Adding a negative literal to a clause.

Adding starting and ending points



(Tentative) Formal Description

Given a 3-SAT formula, construct a directed graph G:

- Nodes of G:
  1. Add node $s = s_0$, $t = t_{n+1}$;
  2. For each variable x_i, add node $s_i = u_{i,0}, t_i = u_{i,m+1}, u_{i,1}, \ldots, u_{i,m}$;
  3. For each clause j, add node v_j.

(Tentative) Formal Description

Given a 3-SAT formula, construct a directed graph G:

- Nodes of G:
  1. Add node $s = s_0$, $t = t_{n+1}$;
  2. For each variable x_i, add node $s_i = u_{i,0}, t_i = u_{i,m+1}, u_{i,1}, \ldots, u_{i,m}$;
  3. For each clause j, add node v_j.
- Edges of G:
  1. For $i = 0, \ldots, n$, connect both s_i and t_i to both s_{i+1} and t_{i+1};

(Tentative) Formal Description

Given a 3-SAT formula, construct a directed graph G:

- Nodes of G:
  1. Add node $s = s_0$, $t = t_{n+1}$;
  2. For each variable x_i, add node $s_i = u_{i,0}, t_i = u_{i,m+1}, u_{i,1}, \ldots, u_{i,m}$;
  3. For each clause j, add node v_j.
- Edges of G:
  1. For $i = 0, \ldots, n$, connect both s_i and t_i to both s_{i+1} and t_{i+1};
  2. For $i = 1, \ldots, n$, $j = 0, 1, \ldots, m$, connect $u_{i,j}$ to $u_{i,j+1}$ and $u_{i,j+1}$ to $u_{i,j}$;

(Tentative) Formal Description

Given a 3-SAT formula, construct a directed graph G:

- Nodes of G:
  1. Add node $s = s_0$, $t = t_{n+1}$;
  2. For each variable x_i, add node $s_i = u_{i,0}, t_i = u_{i,m+1}, u_{i,1}, \ldots, u_{i,m}$;
  3. For each clause j, add node v_j.
- Edges of G:
  1. For $i = 0, \ldots, n$, connect both s_i and t_i to both s_{i+1} and t_{i+1};
  2. For $i = 1, \ldots, n$, $j = 0, 1, \ldots, m$, connect $u_{i,j}$ to $u_{i,j+1}$ and $u_{i,j+1}$ to $u_{i,j}$;
  3. For each clause j that includes literal x_i, connect $u_{i,j}$ to v_j and v_j to $u_{i,j+1}$;

(Tentative) Formal Description

Given a 3-SAT formula, construct a directed graph G:

- Nodes of G:
  1. Add node $s = s_0$, $t = t_{n+1}$;
  2. For each variable x_i, add node $s_i = u_{i,0}, t_i = u_{i,m+1}, u_{i,1}, \ldots, u_{i,m}$;
  3. For each clause j, add node v_j.
- Edges of G:
  1. For $i = 0, \ldots, n$, connect both s_i and t_i to both s_{i+1} and t_{i+1};
  2. For $i = 1, \ldots, n$, $j = 0, 1, \ldots, m$, connect $u_{i,j}$ to $u_{i,j+1}$ and $u_{i,j+1}$ to $u_{i,j}$;
  3. For each clause j that includes literal x_i, connect $u_{i,j}$ to v_j and v_j to $u_{i,j+1}$;
  4. For each cluase j that includes literals $\neg x_i$, connect $u_{i,j+1}$ to v_j and v_j to $u_{i,j}$;

(Tentative) Formal Description

Given a 3-SAT formula, construct a directed graph G:

- Nodes of G:
  1. Add node $s = s_0$, $t = t_{n+1}$;
  2. For each variable x_i, add node $s_i = u_{i,0}, t_i = u_{i,m+1}, u_{i,1}, \ldots, u_{i,m}$;
  3. For each clause j, add node v_j.
- Edges of G:
  1. For $i = 0, \ldots, n$, connect both s_i and t_i to both s_{i+1} and t_{i+1};
  2. For $i = 1, \ldots, n$, $j = 0, 1, \ldots, m$, connect $u_{i,j}$ to $u_{i,j+1}$ and $u_{i,j+1}$ to $u_{i,j}$;
  3. For each clause j that includes literal x_i, connect $u_{i,j}$ to v_j and v_j to $u_{i,j+1}$;
  4. For each cluase j that includes literals $\neg x_i$, connect $u_{i,j+1}$ to v_j and v_j to $u_{i,j}$;
  5. Connect t to s.

(Tentatively) Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

(Tentatively) Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Proof.

Given a satisfying truth assignment, there is a corresponding Hamiltonian cycle in G; (easy direction)

(Tentatively) Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Proof.

Given a satisfying truth assignment, there is a corresponding Hamiltonian cycle in G; (easy direction)
Given a Hamiltonian cycle in G, there is a satisfying truth assignment: we run into a problem!

(Tentatively) Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Proof.

Given a satisfying truth assignment, there is a corresponding Hamiltonian cycle in G; (easy direction)
Given a Hamiltonian cycle in G, there is a satisfying truth assignment: we run into a problem!
What if the cycle, after going from $u_{i,j}$ to v_j, then jumps to $u_{i',j+1}$? $u_{i,j+1}$ may still be covered by clause $j+1$ when the cycle jumps back from v_{j+1}?

(Tentatively) Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Proof.

Given a satisfying truth assignment, there is a corresponding Hamiltonian cycle in G; (easy direction)
Given a Hamiltonian cycle in G, there is a satisfying truth assignment: we run into a problem!
What if the cycle, after going from $u_{i,j}$ to v_j, then jumps to $u_{i',j+1}$? $u_{i,j+1}$ may still be covered by clause $j+1$ when the cycle jumps back from v_{j+1}?
This does not easily correspond to a truth assignment.

(Tentatively) Completing the Proof

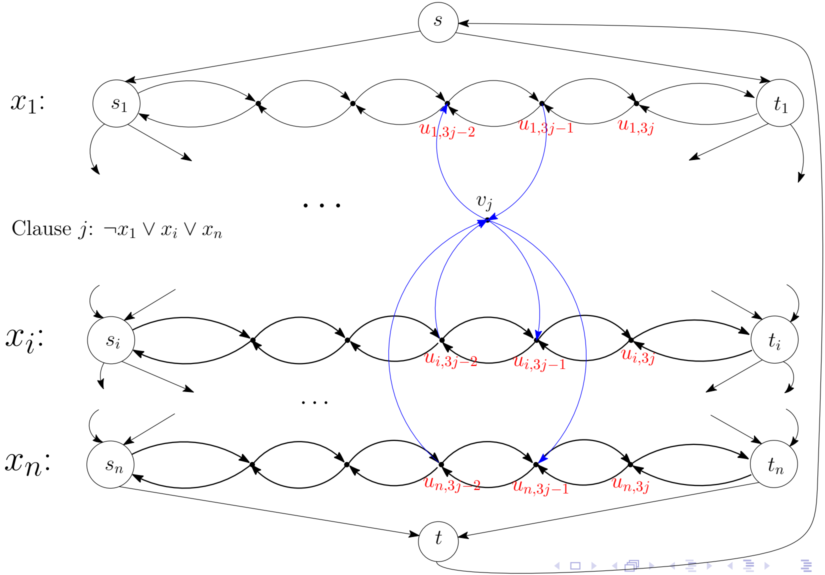
Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Proof.

Given a satisfying truth assignment, there is a corresponding Hamiltonian cycle in G; (easy direction)
Given a Hamiltonian cycle in G, there is a satisfying truth assignment: we run into a problem!
What if the cycle, after going from $u_{i,j}$ to v_j, then jumps to $u_{i',j+1}$? $u_{i,j+1}$ may still be covered by clause $j+1$ when the cycle jumps back from v_{j+1}?
This does not easily correspond to a truth assignment.
Idea: Refine the construction to prevent such jumping from happening.

Refined Construction



Formal Description

Given a 3-SAT formula, construct a directed graph G:

- Nodes of G:
  1. Add node $s = s_0$, $t = t_{n+1}$;
  2. For each variable x_i, add node $s_i = u_{i,0}, t_i = u_{i,3m+1}, u_{i,1}, \ldots, u_{i,3m}$;
  3. For each clause j, add node v_j.
- Edges of G:
  1. For $i = 0, \ldots, n$, connect both s_i and t_i to both s_{i+1} and t_{i+1};
  2. For $i = 1, \ldots, n$, $j = 0, 1, \ldots, 3m$, connect $u_{i,j}$ to $u_{i,j+1}$ and $u_{i,j+1}$ to $u_{i,j}$;
  3. For each clause j that includes literal x_i, connect $u_{i,3j-2}$ to v_j and v_j to $u_{i,3j-1}$;
  4. For each cluase j that includes literals $\neg x_i$, connect $u_{i,3j-1}$ to v_j and v_j to $u_{i,3j-2}$;
  5. Connect t to s.

Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Proof.

Given a satisfying truth assignment, there is a corresponding Hamiltonian cycle in G; (easy direction)

Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Proof.

Given a satisfying truth assignment, there is a corresponding Hamiltonian cycle in G; (easy direction)
Given a Hamiltonian cycle in G, there is a satisfying truth assignment:
Now no jumping around is possible, becuase

Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Proof.

Given a satisfying truth assignment, there is a corresponding Hamiltonian cycle in G; (easy direction)
Given a Hamiltonian cycle in G, there is a satisfying truth assignment:
Now no jumping around is possible, becuase if a Hamiltonian cycle visit c_j from $u_{i,3j-2}$, it has to go back to $u_{i,3j-1}$, otherwise $u_{i,3j-1}$ has only one neighbor ($u_{i,3j}$) left (and therefore can no longer be on a Hamiltonian cycle); the same is true if c_j is visited from $u_{i,3j-1}$.

Completing the Proof

Claim

The given 3-SAT formula has a satisfying truth assignment if and only if G has a Hamiltonian cycle.

Proof.

Given a satisfying truth assignment, there is a corresponding Hamiltonian cycle in G; (easy direction)
Given a Hamiltonian cycle in G, there is a satisfying truth assignment:
Now no jumping around is possible, becuase if a Hamiltonian cycle visit c_j from $u_{i,3j-2}$, it has to go back to $u_{i,3j-1}$, otherwise $u_{i,3j-1}$ has only one neighbor ($u_{i,3j}$) left (and therefore can no longer be on a Hamiltonian cycle); the same is true if c_j is visited from $u_{i,3j-1}$.
The way each x_i cycle is traversed by the Hamiltonian cycle now corresponds to a truth assignment. It is straightforward to verify that all clauses are satisfied by this assignment. □

Hamiltonian Paths

Proposition

HAMILTONIAN CYCLE $\leq_P$ HAMILTONIAN PATH.

Hamiltonian Paths

Proposition

HAMILTONIAN CYCLE $\leq_P$ HAMILTONIAN PATH.

Proof Sketch.

Pick one node and split it into two copies, one with all incoming edges and the other with all outgoing edges. □

Eulerian Tours and Eulerian Circuits

Definition

Given an undirected graph G, a *Eulerian trail* (or Eulerian path) is a path that traverses each *edge* in G exactly once. A Eulerian trail that is a cycle is called a Eulerian circuit (or Eulerian cycle).

Eulerian Tours and Eulerian Circuits

Definition

Given an undirected graph G, a *Eulerian trail* (or Eulerian path) is a path that traverses each *edge* in G exactly once. A Eulerian trail that is a cycle is called a Eulerian circuit (or Eulerian cycle).

Remark

Note that a Eulerian trail does not need to be simple.

Eulerian Tours and Eulerian Circuits

Definition

Given an undirected graph G, a *Eulerian trail* (or Eulerian path) is a path that traverses each *edge* in G exactly once. A Eulerian trail that is a cycle is called a Eulerian circuit (or Eulerian cycle).

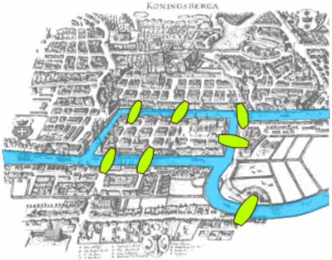
Remark

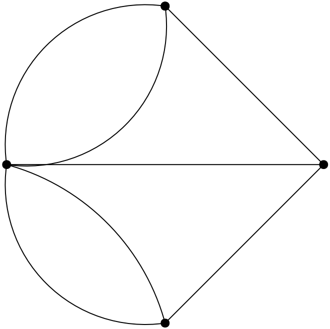
Note that a Eulerian trail does not need to be simple.

Theorem

Deciding whether a given graph has a Eulerian cycle is in P*.*
Deciding whether a given graph has a Eulerian path is in P*.*

Seven Bridges of Königsberg





Euler's theorem

Theorem

A graph G has a Eulerian cycle if and only if all its nodes have even degrees and all the nodes with non-zero degrees are in the same connected component.

Euler's theorem

Theorem

A graph G has a Eulerian cycle if and only if all its nodes have even degrees and all the nodes with non-zero degrees are in the same connected component.

Proof by induction.

Euler's theorem

Theorem

A graph G has a Eulerian cycle if and only if all its nodes have even degrees and all the nodes with non-zero degrees are in the same connected component.

Proof by induction.

Theorem

A graph G has a Eulerian path if and only if exactly zero or two of its nodes have odd degrees, and all the nodes with non-zero degrees are in the same connected component.

Euler's theorem

Theorem

A graph G has a Eulerian cycle if and only if all its nodes have even degrees and all the nodes with non-zero degrees are in the same connected component.

Proof by induction.

Theorem

A graph G has a Eulerian path if and only if exactly zero or two of its nodes have odd degrees, and all the nodes with non-zero degrees are in the same connected component.

Proof sketch for a graph with two odd-degree nodes: First find a path from one odd degree node to the other, then the rest of the graph has a Eulerian cycle in each of its connected component. Concatenate these cycles to the path forms a Eulerian path.

Traveling Salesman Problem

Definition

Given n nodes $v_1, \ldots, v_n$, a *tour* is a path that starts from v_1, visits every other node exactly once, and returns to v_1.
In the *Traveling Salesman Problem (TSP)*, we are given n nodes and a distance $d_{i,j}$ from each node v_i to another one v_j, and a bound D. We are asked to decide whether there is a tour of total distance at most D.

Traveling Salesman Problem

Definition

Given n nodes $v_1, \ldots, v_n$, a *tour* is a path that starts from v_1, visits every other node exactly once, and returns to v_1.
In the *Traveling Salesman Problem (TSP)*, we are given n nodes and a distance $d_{i,j}$ from each node v_i to another one v_j, and a bound D. We are asked to decide whether there is a tour of total distance at most D.

Proposition

HAMILTONIAN CYCLE $\leq_P$ TSP. $\Rightarrow$ TSP is NP-complete.

Traveling Salesman Problem

Definition

Given n nodes $v_1, \ldots, v_n$, a *tour* is a path that starts from v_1, visits every other node exactly once, and returns to v_1.
In the *Traveling Salesman Problem (TSP)*, we are given n nodes and a distance $d_{i,j}$ from each node v_i to another one v_j, and a bound D. We are asked to decide whether there is a tour of total distance at most D.

Proposition

HAMILTONIAN CYCLE $\leq_P$ TSP. $\Rightarrow$ TSP is NP-complete.

Proof.

Easy to see TSP $\in$ NP.

Traveling Salesman Problem

Definition

Given n nodes $v_1, \ldots, v_n$, a *tour* is a path that starts from v_1, visits every other node exactly once, and returns to v_1.
In the *Traveling Salesman Problem (TSP)*, we are given n nodes and a distance $d_{i,j}$ from each node v_i to another one v_j, and a bound D. We are asked to decide whether there is a tour of total distance at most D.

Proposition

HAMILTONIAN CYCLE $\leq_P$ TSP. $\Rightarrow$ TSP is NP-complete.

Proof.

Easy to see TSP $\in$ NP. Given an instance of HAMILTONIAN CYCLE $G = (V, E)$, construct a TSP instance: the nodes are V; distance $d_{i,j} = 1$ if $(v_i, v_j) \in E$, and 2 otherwise.

Traveling Salesman Problem

Definition

Given n nodes $v_1, \dots, v_n$, a *tour* is a path that starts from v_1, visits every other node exactly once, and returns to v_1.
In the *Traveling Salesman Problem (TSP)*, we are given n nodes and a distance $d_{i,j}$ from each node v_i to another one v_j, and a bound D. We are asked to decide whether there is a tour of total distance at most D.

Proposition

HAMILTONIAN CYCLE $\leq_P$ TSP. $\Rightarrow$ TSP is NP-complete.

Proof.

Easy to see TSP $\in$ NP. Given an instance of HAMILTONIAN CYCLE $G = (V, E)$, construct a TSP instance: the nodes are V; distance $d_{i,j} = 1$ if $(v_i, v_j) \in E$, and 2 otherwise. A Hamiltonian cycle exists in G if and only if there is a tour in the TSP instance with distance n. □

Last remarks on TSP

Remark: When $d_{i,j} = d_{j,i}$, $\forall i, j$, the problem is called a *symmetric TSP* problem; otherwise it is said to be asymmetric.

Last remarks on TSP

Remark: When $d_{i,j} = d_{j,i}$, $\forall i, j$, the problem is called a *symmetric TSP* problem; otherwise it is said to be asymmetric.
When $d_{i,j} + d_{j,k} \geq d_{i,k}$, $\forall i, j, k$, the problem is called a *metric TSP* problem.

Last remarks on TSP

Remark: When $d_{i,j} = d_{j,i}$, $\forall i, j$, the problem is called a *symmetric TSP* problem; otherwise it is said to be asymmetric.
When $d_{i,j} + d_{j,k} \geq d_{i,k}$, $\forall i, j, k$, the problem is called a *metric TSP* problem.
We showed that asymmetric, metric TSP is NP-complete.

Last remarks on TSP

Remark: When $d_{i,j} = d_{j,i}$, $\forall i, j$, the problem is called a *symmetric TSP* problem; otherwise it is said to be asymmetric.
When $d_{i,j} + d_{j,k} \geq d_{i,k}$, $\forall i, j, k$, the problem is called a *metric TSP* problem.
We showed that asymmetric, metric TSP is NP-complete.
In fact, symmetric, metric TSP is already NP-complete.

Exercises

Does the following problem admit a polynomial-time algorithm or is it NP-complete?

1. Given a set $A = \{a_1, \dots, a_n\}$, a collection $B_1, B_2, \cdots, B_m$ of subsets of A, and an integer $k > 0$. Is there a set $H \subseteq A$, $|H| \leq k$ such that $H \cap B_i \neq \emptyset$ for $i = 1, \dots, m$?

Exercises

Does the following problem admit a polynomial-time algorithm or is it NP-complete?

1. Given a set $A = \{a_1, \ldots, a_n\}$, a collection $B_1, B_2, \cdots, B_m$ of subsets of A, and an integer $k > 0$. Is there a set $H \subseteq A$, $|H| \leq k$ such that $H \cap B_i \neq \emptyset$ for $i = 1, \ldots, m$?
2. Given a directed graph $G = (V, E)$ with $s, t \in V$, and an integer $k > 0$.
    1. Does there exist at least k edge-disjoint paths from s to t?

Exercises

Does the following problem admit a polynomial-time algorithm or is it NP-complete?

1. Given a set $A = \{a_1, \dots, a_n\}$, a collection $B_1, B_2, \cdots, B_m$ of subsets of A, and an integer $k > 0$. Is there a set $H \subseteq A$, $|H| \leq k$ such that $H \cap B_i \neq \emptyset$ for $i = 1, \dots, m$?
2. Given a directed graph $G = (V, E)$ with $s, t \in V$, and an integer $k > 0$.
   1. Does there exist at least k edge-disjoint paths from s to t?
   2. Given m paths $P_1, \cdots, P_m$ from s to t, does there exist at least k paths among $P_1, \cdots, P_m$ that are edge-disjoint?

Exercises

1. Given an undirected graph and integer k, decide whether there is a spanning tree in which each node has degree at most k.

Exercises

1. Given an undirected graph and integer k, decide whether there is a spanning tree in which each node has degree at most k.
2. Given a directed graph $G = (V, E)$ with $s, t \in V$ and nonnegative integral edge weights, and an integer $k > 0$.
   1. Does there exist a simple path from s to t with total weight at most k?

Exercises

1. Given an undirected graph and integer k, decide whether there is a spanning tree in which each node has degree at most k.
2. Given a directed graph $G = (V, E)$ with $s, t \in V$ and nonnegative integral edge weights, and an integer $k > 0$.
   1. Does there exist a simple path from s to t with total weight at most k?
   2. Does there exist a simple path from s to t with total weight at least k?