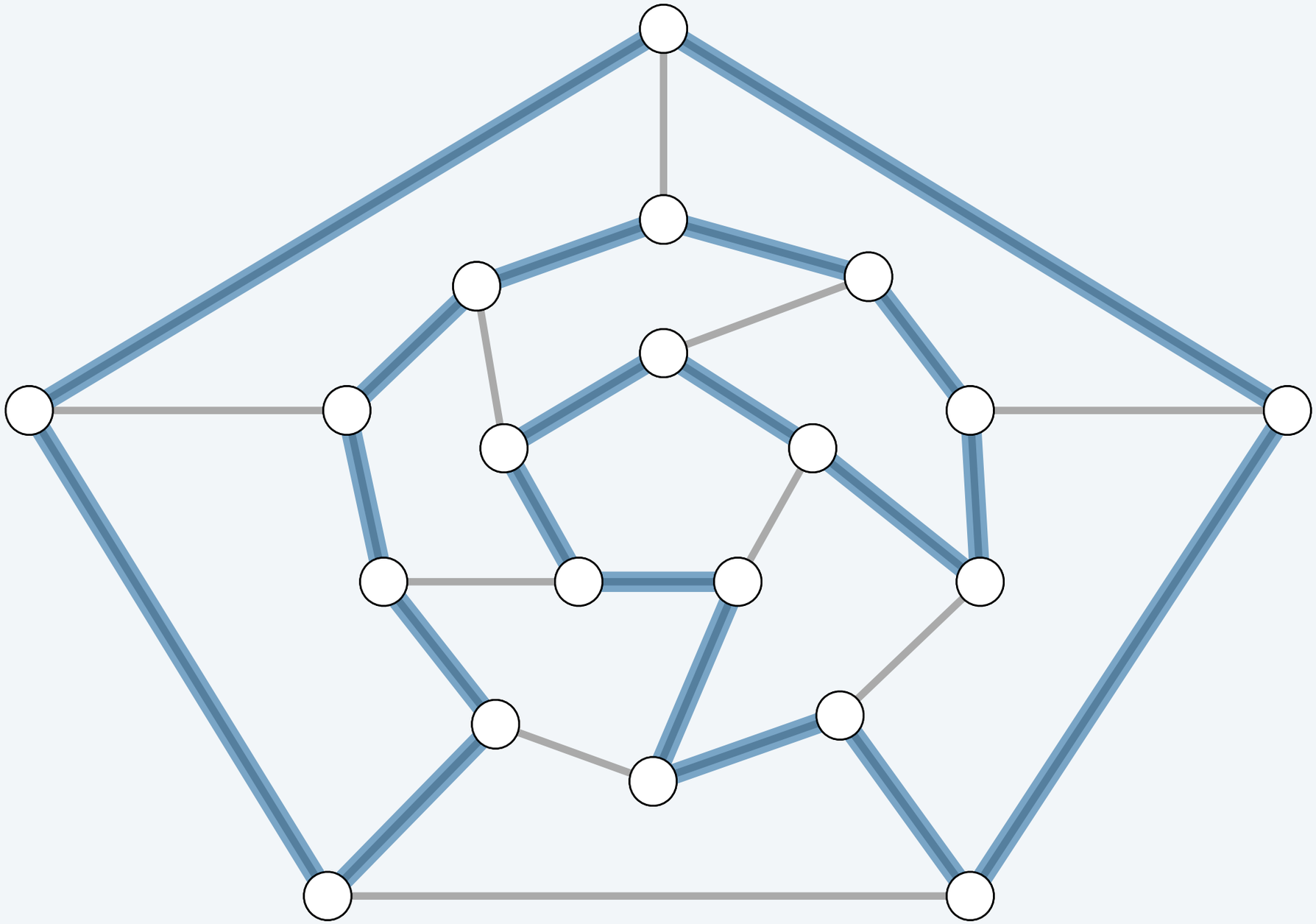
INTRACTABILITY III

- *special cases: trees*
- *special cases: planarity*
- *approximation algorithms: vertex cover*
- *approximation algorithms: knapsack*
- *exponential algorithms: 3-SAT*
- *exponential algorithms: TSP*

Hamilton cycle

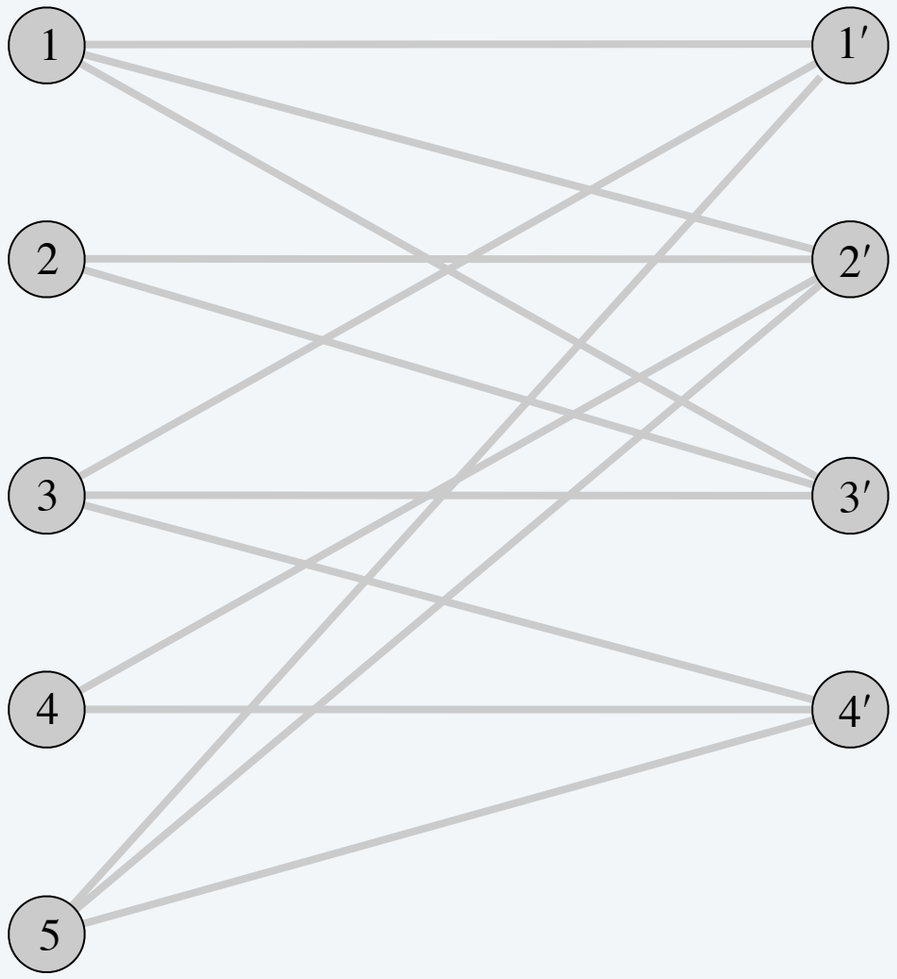
HAMILTON-CYCLE. Given an undirected graph $G = (V, E)$, does there exist a cycle Γ that visits every node exactly once?



yes

Hamilton cycle

HAMILTON-CYCLE. Given an undirected graph $G = (V, E)$, does there exist a cycle Γ that visits every node exactly once?



no

Trails, Paths, and Circuits

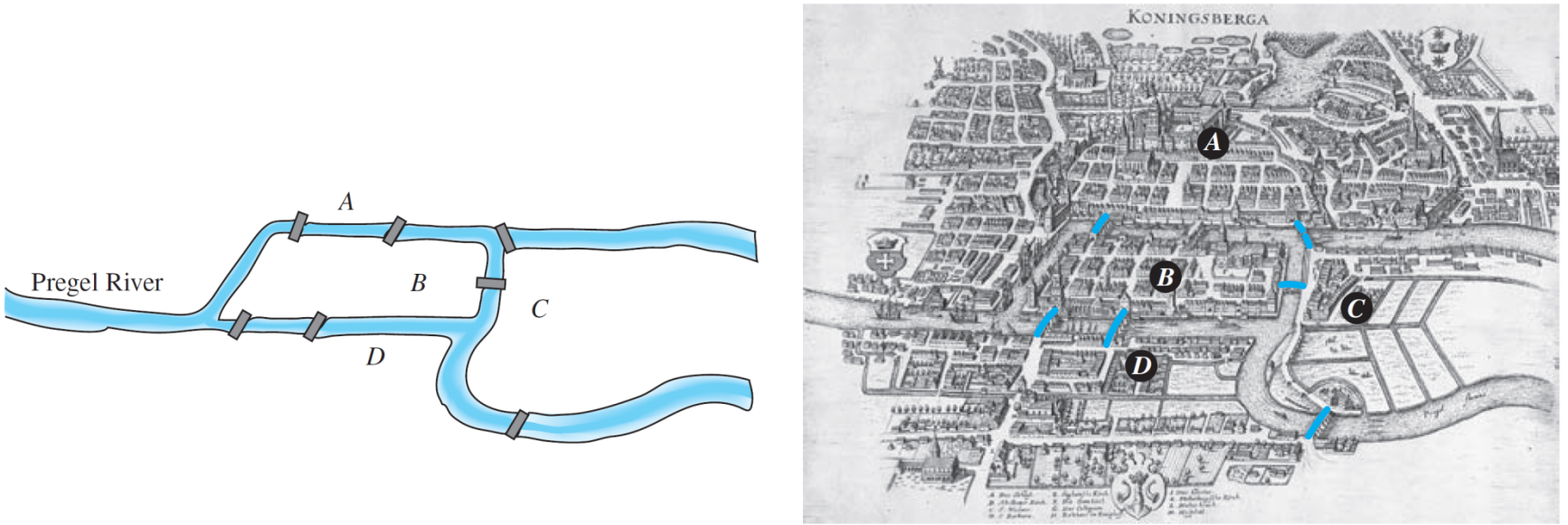
The subject of graph theory began in the year 1736 when the great mathematician Leonhard Euler published a paper giving the solution to the following puzzle:

The town of Königsberg in Prussia (now Kaliningrad in Russia) was built at a point where two branches of the Pregel River came together.

It consisted of an island and some land along the river banks.

Trails, Paths, and Circuits

These were connected by seven bridges as shown in Figure 10.2.1.



The Seven Bridges of Königsberg

Figure 10.2.1

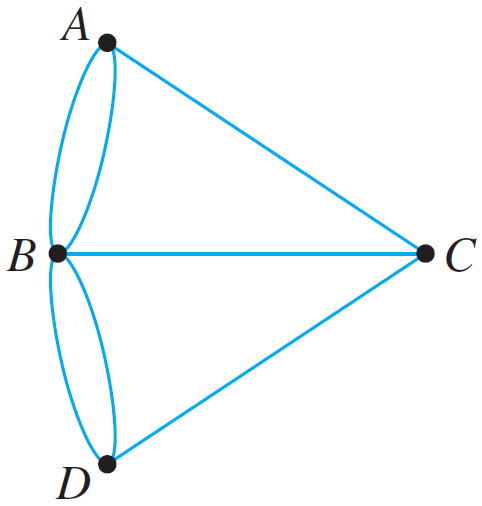
Trails, Paths, and Circuits

The question is this: Is it possible for a person to take a walk around town, starting and ending at the same location and crossing each of the seven bridges exactly once?

To solve this puzzle, Euler translated it into a graph theory problem. He noticed that all points of a given land mass can be identified with each other since a person can travel from any one point to any other point of the same land mass without crossing a bridge.

Trails, Paths, and Circuits

Thus for the purpose of solving the puzzle, the map of Königsberg can be identified with the graph shown in Figure 10.2.2, in which the vertices *A*, *B*, *C*, and *D* represent land masses and the seven edges represent the seven bridges.



Graph Version of Königsberg Map

Figure 10.2.2

Trails, Paths, and Circuits

In terms of this graph, the question becomes the following:

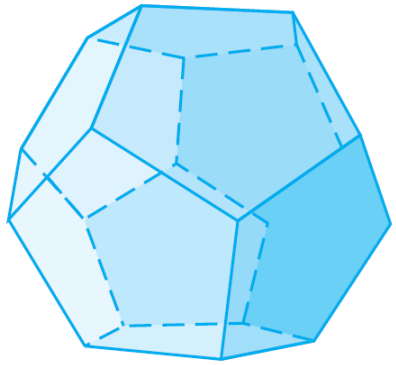
Is it possible to find a route through the graph that starts and ends at some vertex, one of *A*, *B*, *C*, or *D*, and traverses each edge exactly once?

Equivalently:

Is it possible to trace this graph, starting and ending at the same point, without ever lifting your pencil from the paper?

Hamiltonian Circuits

In 1859 the Irish mathematician Sir William Rowan Hamilton introduced a puzzle in the shape of a dodecahedron (a solid figure with 12 identical pentagonal faces.)



Dodecahedron

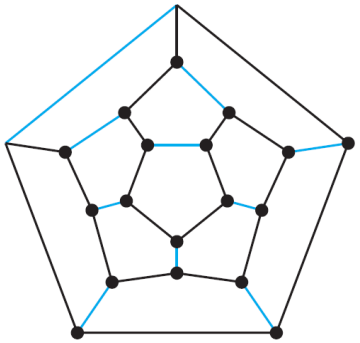
Figure 10.2.6

Hamiltonian Circuits

Each vertex was labeled with the name of a city—London, Paris, Hong Kong, New York, and so on.

The problem Hamilton posed was to start at one city and tour the world by visiting each other city exactly once and returning to the starting city.

One way to solve the puzzle is to imagine the surface of the dodecahedron stretched out and laid flat in the plane, as follows:



Hamiltonian Circuits

The circuit denoted with black lines is one solution. Note that although every city is visited, many edges are omitted from the circuit. (More difficult versions of the puzzle required that certain cities be visited in a certain order.)

The following definition is made in honor of Hamilton.

• Definition

Given a graph G, a **Hamiltonian circuit** for G is a simple circuit that includes every vertex of G. That is, a Hamiltonian circuit for G is a sequence of adjacent vertices and distinct edges in which every vertex of G appears exactly once, except for the first and the last, which are the same.

Hamiltonian Circuits

There is, however, a simple technique that can be used in many cases to show that a graph does *not* have a Hamiltonian circuit.

This follows from the following considerations:

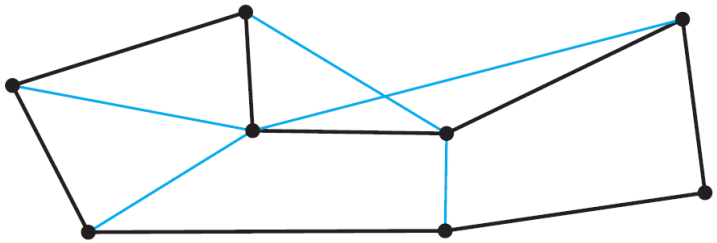
Suppose a graph G with at least two vertices has a Hamiltonian circuit C given concretely as

$$C\colon v_0 e_1 v_1 e_2 \cdots v_{n-1} e_n v_n.$$

Since C is a simple circuit, all the e_i are distinct and all the v_j are distinct except that $v_0 = v_n$. Let H be the subgraph of G that is formed using the vertices and edges of C.

Hamiltonian Circuits

An example of such an H is shown below.



H is indicated by the black lines.

Note that H has the same number of edges as it has vertices since all its n edges are distinct and so are its n vertices $v_1, v_2, \ldots, v_n$.

Also, by definition of Hamiltonian circuit, every vertex of G is a vertex of H, and H is connected since any two of its vertices lie on a circuit. In addition, every vertex of H has degree 2.

Hamiltonian Circuits

The reason for this is that there are exactly two edges incident on any vertex. These are e_i and e_{i+1} for any vertex v_i except $v_0 = v_n$, and they are e_1 and e_n for v_0 (= v_n).

These observations have established the truth of the following proposition in all cases where G has at least two vertices.

Proposition 10.2.6

If a graph G has a Hamiltonian circuit, then G has a subgraph H with the following properties:

1. H contains every vertex of G.
2. H is connected.
3. H has the same number of edges as vertices.
4. Every vertex of H has degree 2.

3-satisfiability reduces to directed Hamilton cycle

Theorem. 3-SAT $\leq_P$ DIRECTED-HAMILTON-CYCLE.

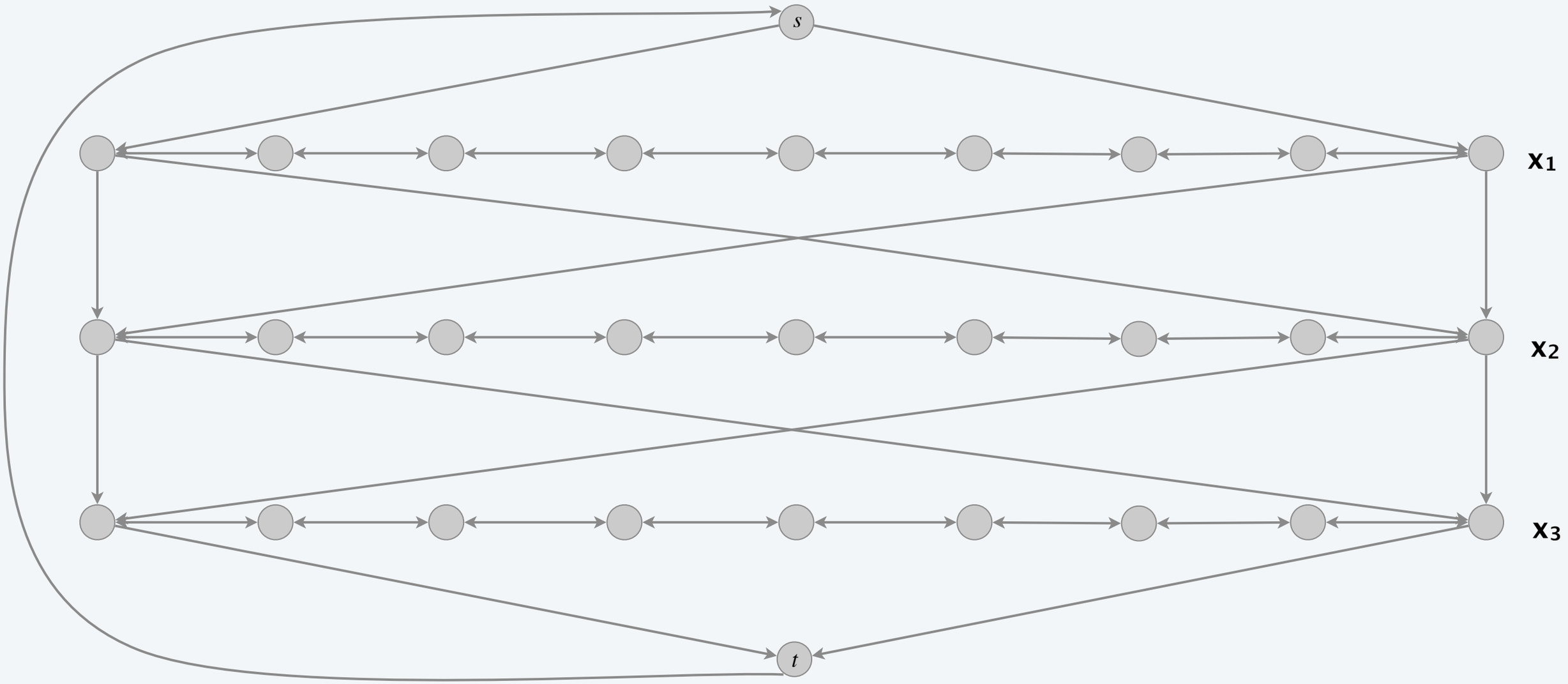
Pf. Given an instance Φ of 3-SAT, we construct an instance G of DIRECTED-HAMILTON-CYCLE that has a Hamilton cycle iff Φ is satisfiable.

Construction overview. Let n denote the number of variables in Φ.
We will construct a graph G that has 2^n Hamilton cycles, with each cycle corresponding to one of the 2^n possible truth assignments.

3-satisfiability reduces to directed Hamilton cycle

Construction. Given 3-SAT instance Φ with n variables x_i and k clauses.

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



Intractability: quiz 5

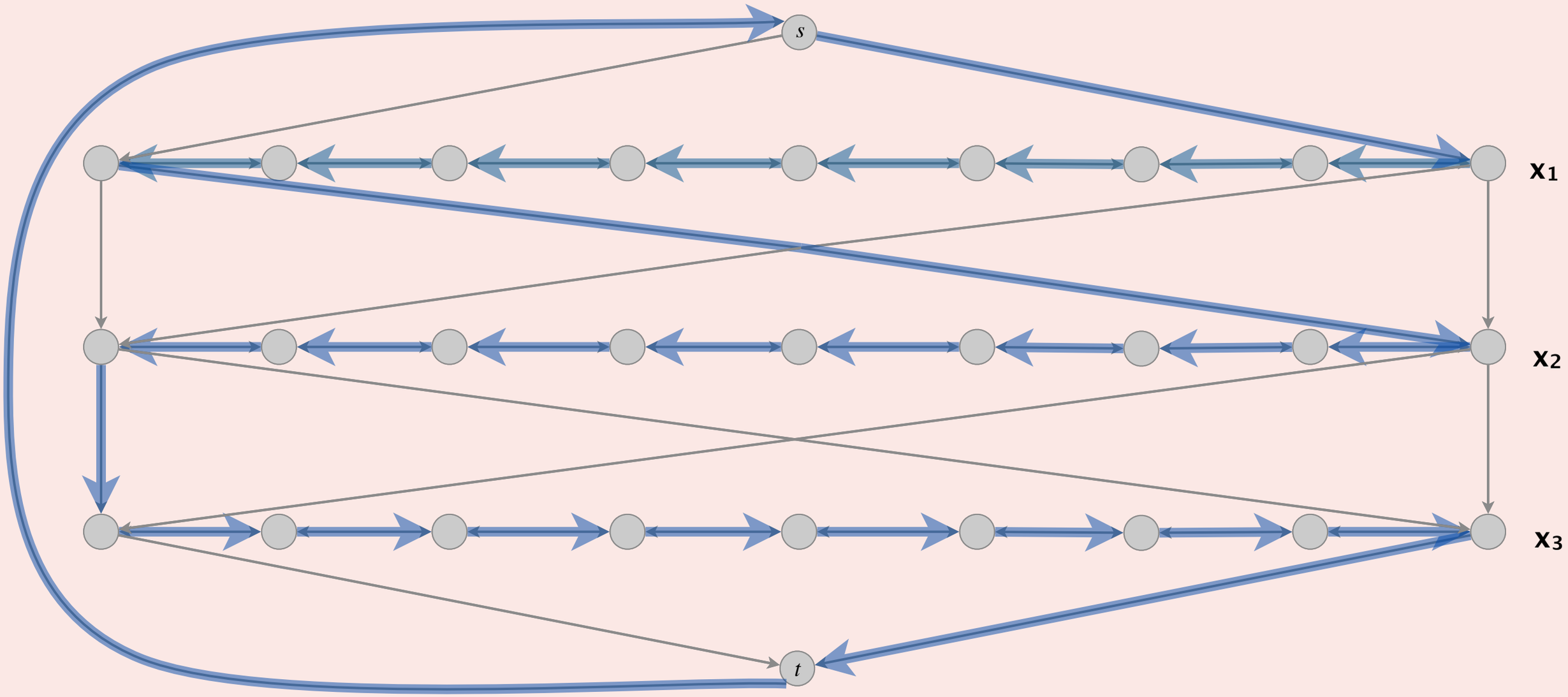
Which is truth assignment corresponding to Hamilton cycle below?

A. $x_1 = true, x_2 = true, x_3 = true$

B. $x_1 = true, x_2 = true, x_3 = false$

C. $x_1 = false, x_2 = false, x_3 = true$

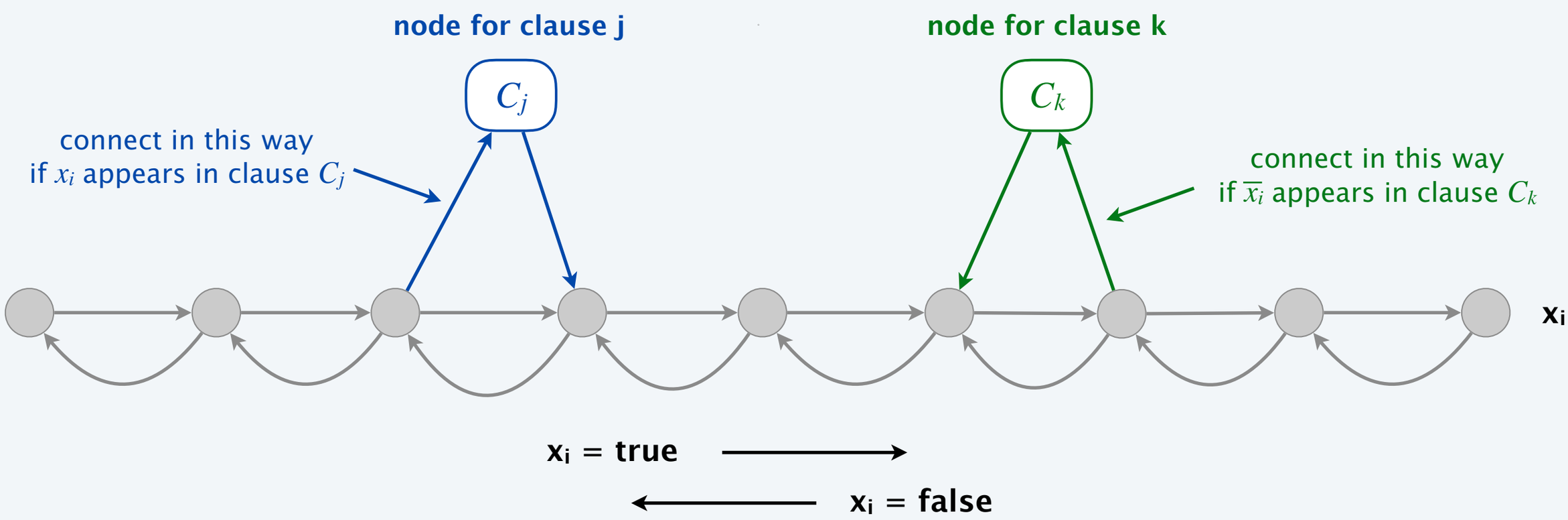
D. $x_1 = false, x_2 = false, x_3 = false$



3-satisfiability reduces to directed Hamilton cycle

Construction. Given 3-SAT instance Φ with n variables x_i and k clauses.

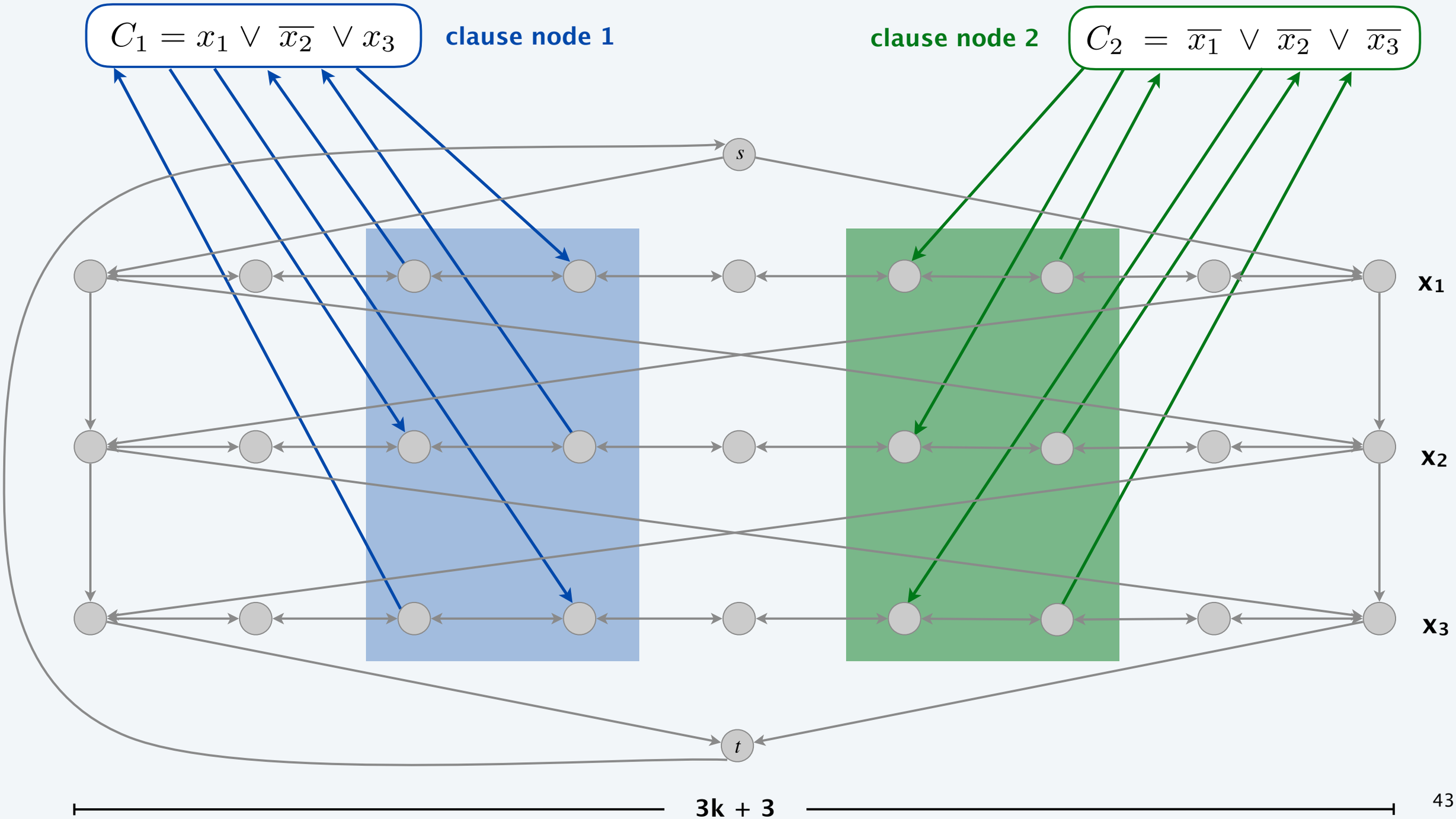
- For each clause: add a node and 2 edges per literal.



3-satisfiability reduces to directed Hamilton cycle

Construction. Given 3-SAT instance Φ with n variables x_i and k clauses.

- For each clause: add a node and 2 edges per literal.



3-satisfiability reduces to directed Hamilton cycle

Lemma. Φ is satisfiable iff G has a Hamilton cycle.

Pf. $\Rightarrow$

- Suppose 3-SAT instance Φ has satisfying assignment x^*.
- Then, define Hamilton cycle Γ in G as follows:
  - if $x_i^* = true$, traverse row i from left to right
  - if $x_i^* = false$, 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 clause node C_j into cycle (and we splice in C_j exactly once) ▪

3-satisfiability reduces to directed Hamilton cycle

Lemma. Φ is satisfiable iff G has a Hamilton cycle.

Pf. $\Leftarrow$

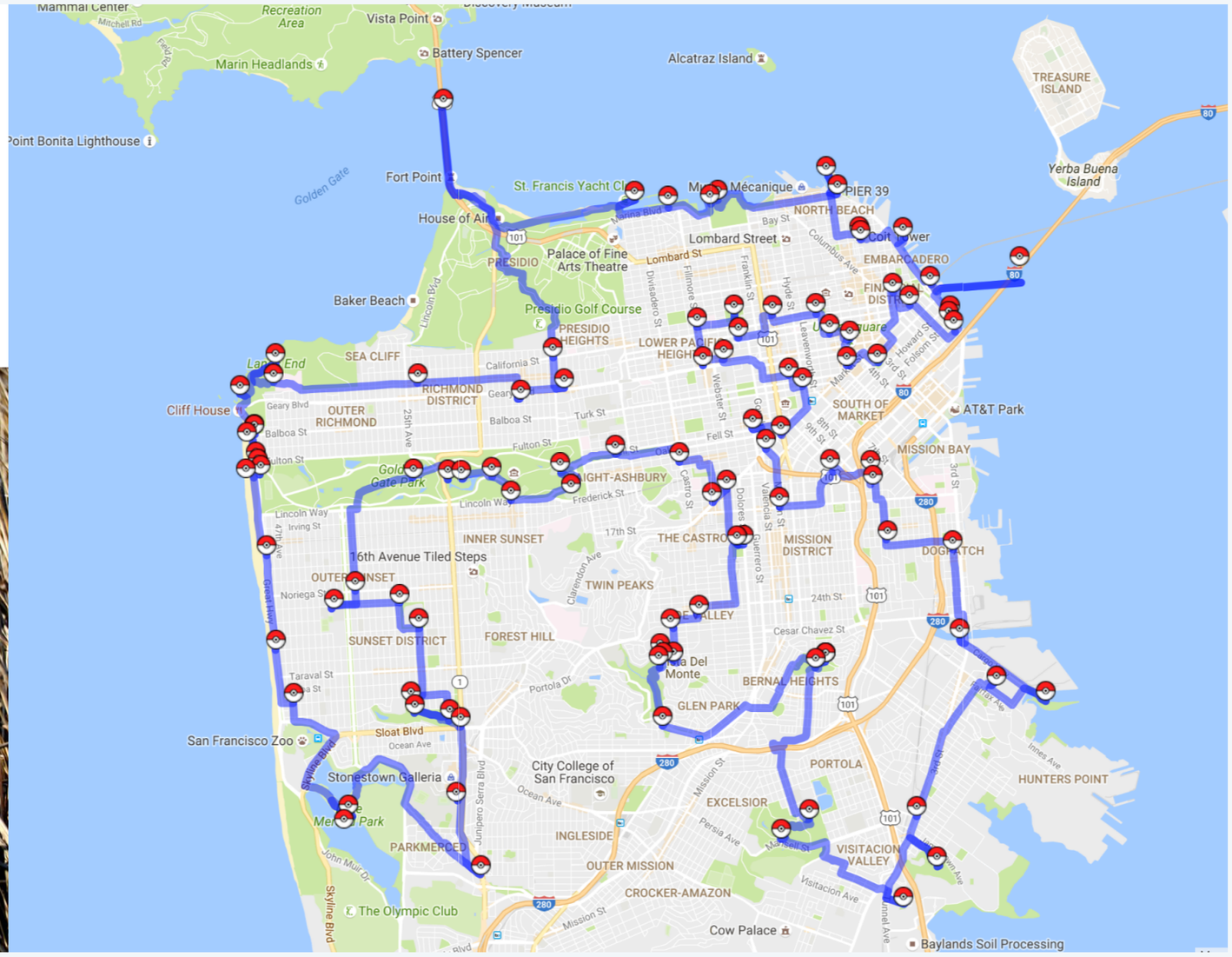
- Suppose G has a Hamilton cycle Γ.
- If Γ enters clause node C_j, it must depart on mate edge.
  - nodes immediately before and after C_j are connected by an edge $e \in E$
  - removing C_j from cycle, and replacing it with edge e yields Hamilton cycle on $G - \{ C_j \}$
- Continuing in this way, we are left with a Hamilton cycle Γ' in $G - \{ C_1, C_2, \ldots, C_k \}$.
- Set $x^{*i} = true$ if Γ' traverses row i left-to-right; otherwise, set $x^{*i} = false$.
- Since the original path visited all clauses at least one of the path was traversed in "correct" direction realative to a clause, and thus each clause is satisfied. ▪

Pokemon Go

Given the locations of n Pokémon, find shortest tour to collect them all.

Map: Where to catch 123 Pokémon in San Francisco

BY ADAM BRINKLOW | OCT 4, 2016, 6:33AM PDT



Traveling salesperson problem

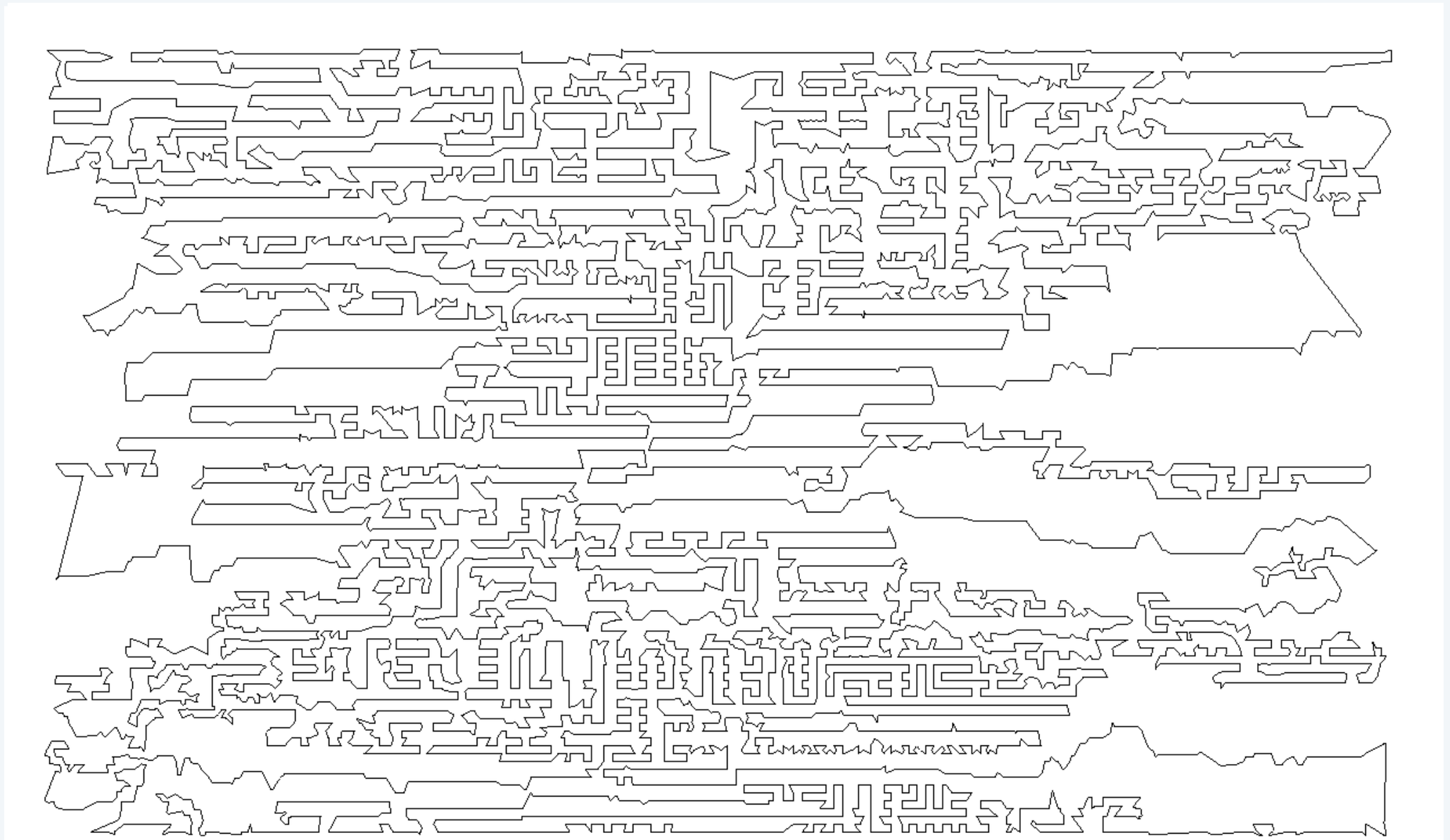
TSP. Given a set of n cities and a pairwise distance function $d(u, v)$, is there a tour of length $\leq D$?

can view as a complete graph

13,509 cities in the United States
http://www.math.uwaterloo.ca/tsp

Traveling salesperson problem

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

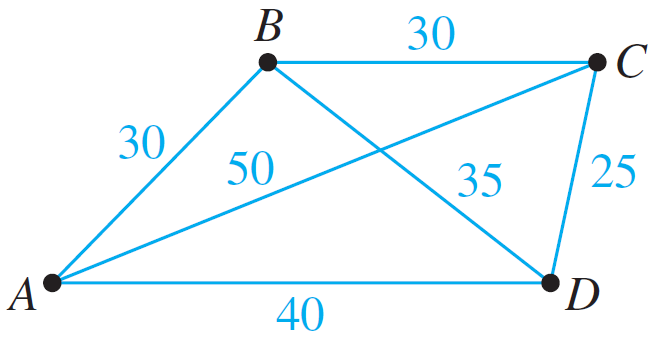


11,849 holes to drill in a programmed logic array
http://www.math.uwaterloo.ca/tsp

Example 9 – *A Traveling Salesman Problem*

Imagine that the drawing below is a map showing four cities and the distances in kilometers between them.

Suppose that a salesman must travel to each city exactly once, starting and ending in city *A*. Which route from city to city will minimize the total distance that must be traveled?



Example 9 – *Solution*

This problem can be solved by writing all possible Hamiltonian circuits starting and ending at *A* and calculating the total distance traveled for each.

Route	Total Distance (In Kilometers)	
$ABCDA$	$30 + 30 + 25 + 40 = 125$	
$ABDCA$	$30 + 35 + 25 + 50 = 140$	
$ACBDA$	$50 + 30 + 35 + 40 = 155$	
$ACDBA$	140	[$ABDCA$ backwards]
$ADBCA$	155	[$ACBDA$ backwards]
$ADCBA$	125	[$ABCDA$ backwards]

Thus either route *ABCDA* or *ADCBA* gives a minimum total distance of 125 kilometers.

Hamiltonian Circuits

The general traveling salesman problem involves finding a Hamiltonian circuit to minimize the total distance traveled for an arbitrary graph with *n* vertices in which each edge is marked with a distance.

One way to solve the general problem is to write down all Hamiltonian circuits starting and ending at a particular vertex, compute the total distance for each, and pick one for which this total is minimal.

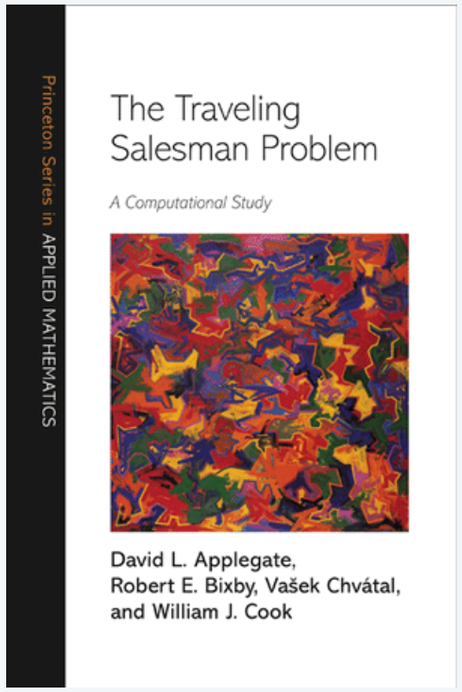
Hamiltonian Circuits

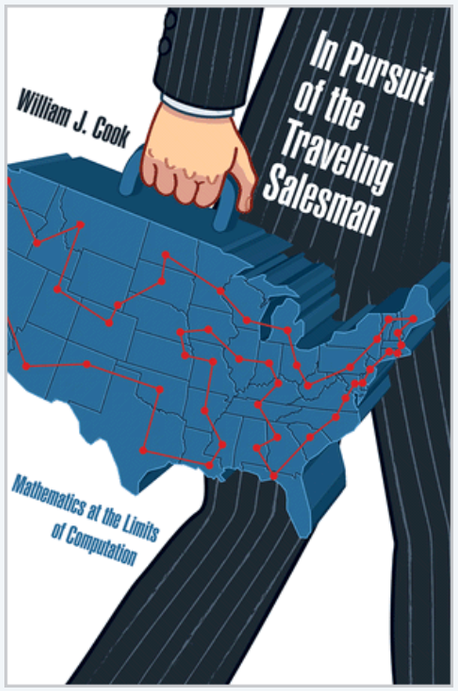
However, even for medium-sized values of n this method is impractical!

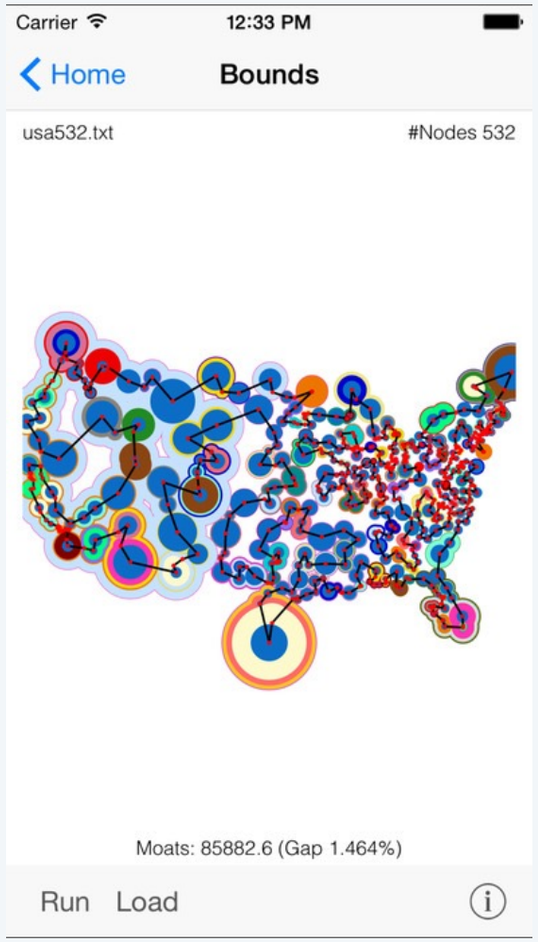
For a complete graph with 30 vertices, there would be $(29!)/2 \cong 4.42 \times 10^{30}$ Hamiltonian circuits starting and ending at a particular vertex to check.

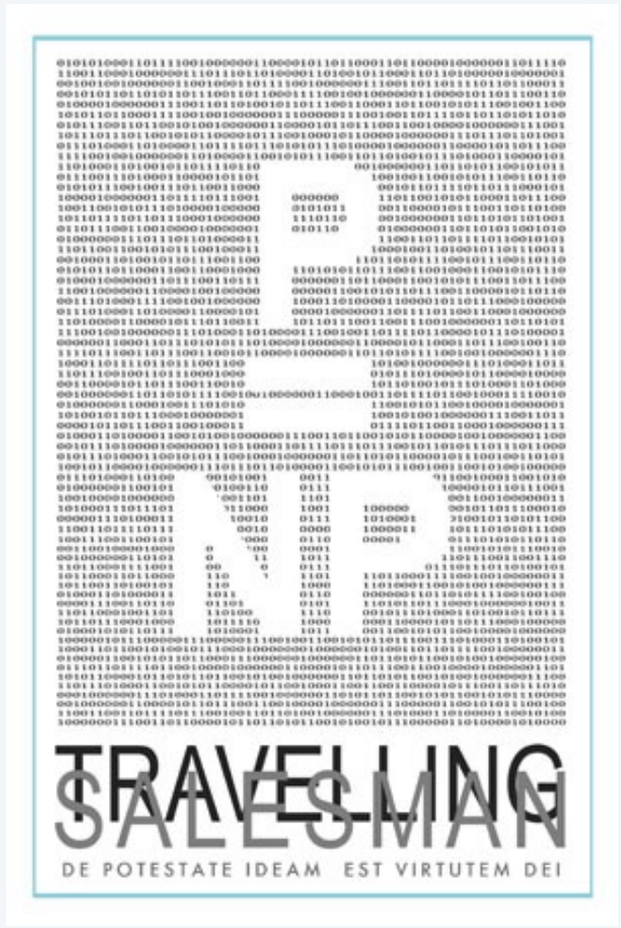
Even if each circuit could be found and its total distance computed in just one nanosecond, it would require approximately **1.4×10^{14} years** to finish the computation.

TSP books, apps, and movies









Hamilton cycle reduces to traveling salesperson problem

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

HAMILTON-CYCLE. Given an undirected graph $G = (V, E)$, does there exist a cycle that visits every node exactly once?

Theorem. HAMILTON-CYCLE $\leq_P$ TSP.

Pf.

- Given an instance $G = (V, E)$ of HAMILTON-CYCLE, create $n = |V|$ 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 has a Hamilton cycle. ▪

Intractability III: quiz 4

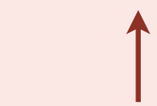
What is complexity of TSP? Choose the best answer.

A. $O(n^2)$

B. $O^*(1.657^n)$

C. $O^*(2^n)$

D. $O^*(n!)$



O^* hides $poly(n)$ terms

Exponential algorithm for TSP: dynamic programming

Theorem. [Held–Karp, Bellman 1962] TSP can be solved in $O(n^2\, 2^n)$ time.

HAMILTON-CYCLE is a special case

J. Soc. Indust. Appl. Math.
Vol. 10, No. 1, March, 1962
Printed in U.S.A.

A DYNAMIC PROGRAMMING APPROACH TO SEQUENCING PROBLEMS*

MICHAEL HELD† AND RICHARD M. KARP†

INTRODUCTION

Many interesting and important optimization problems require the determination of a best order of performing a given set of operations. This paper is concerned with the solution of three such *sequencing problems*: a scheduling problem involving arbitrary cost functions, the traveling-salesman problem, and an assembly-line balancing problem. Each of these problems has a structure permitting solution by means of recursion schemes of the type associated with dynamic programming. In essence, these recursion schemes permit the problems to be treated in terms of *combinations*, rather than *permutations*, of the operations to be performed. The dynamic programming formulations are given in §1, together with a discussion of various extensions such as the inclusion of precedence constraints. In each case the proposed method of solution is computationally effective for problems in a certain limited range. Approximate solutions to larger problems may be obtained by solving sequences of small derived problems, each having the same structure as the original one. This procedure of successive approximations is developed in detail in §2 with specific reference to the traveling-salesman problem, and §3 summarizes computational experience with an IBM 7090 program using the procedure.

Dynamic Programming Treatment of the Travelling Salesman Problem*

RICHARD BELLMAN

RAND Corporation, Santa Monica, California

Introduction

The well-known travelling salesman problem is the following: "A salesman is required to visit once and only once each of n different cities starting from a base city, and returning to this city. What path minimizes the total distance travelled by the salesman?"

The problem has been treated by a number of different people using a variety of techniques; cf. Dantzig, Fulkerson, Johnson [1], where a combination of ingenuity and linear programming is used, and Miller, Tucker and Zemlin [2], whose experiments using an all-integer program of Gomory did not produce results in cases with ten cities although some success was achieved in cases of simply four cities. The purpose of this note is to show that this problem can easily be formulated in dynamic programming terms [3], and resolved computationally for up to 17 cities. For larger numbers, the method presented below, combined with various simple manipulations, may be used to obtain quick approximate solutions. Results of this nature were independently obtained by M. Held and R. M. Karp, who are in the process of publishing some extensions and computational results.

Exponential algorithm for TSP: dynamic programming

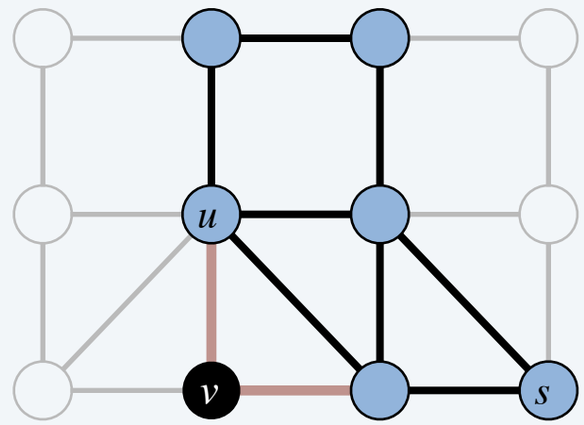
Theorem. [Held–Karp, Bellman 1962] TSP can be solved in $O(n^2\, 2^n)$ time.

Pf. [dynamic programming]

pick node s arbitrarily

- Subproblems: $c(s, v, X)$ = cost of cheapest path between s and $v \neq s$ that visits every node in X exactly once (and uses only nodes in X).
- Goal: $\min_{v \in V} c(s, v, V) + c(v, s)$
- There are $\leq n\, 2^n$ subproblems and they satisfy the recurrence:

$$c(s, v, X) = \begin{cases} c(s, v) & \text{if } |X| = 2 \\ \min_{u \in X \setminus \{s, v\}} c(s, u, X \setminus \{v\}) + c(u, v) & \text{if } |X| > 2. \end{cases}$$



- The values $c(s, v, X)$ can be computed in increasing order of the cardinality of X. ▪

22-city TSP instance takes 1,000 years

The Washington Post

Quantum computers are straight out of science fiction. Take the "traveling salesman problem," where a salesperson has to visit a specific set of cities, each only once, and return to the first city by the most efficient route possible. As the number of cities increases, the problem becomes exponentially complex. It would take a laptop computer 1,000 years to compute the most efficient route between 22 cities, for example. A quantum computer could do this within minutes, possibly seconds.

$2^{22} = 4{,}194{,}304$

$22! = 1{,}124{,}000{,}727{,}777{,}607{,}680{,}000 \sim 10^{21}$

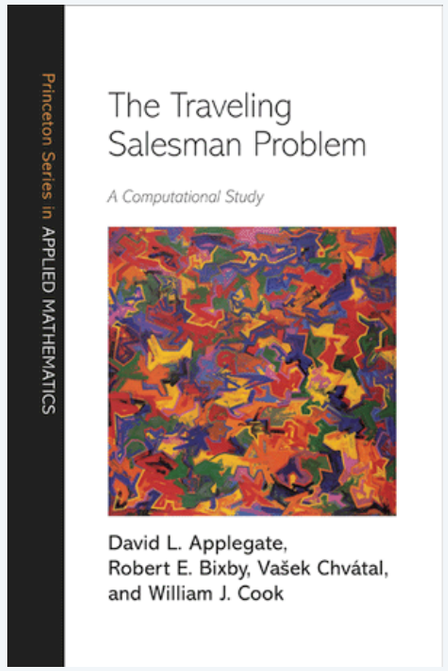
Concorde TSP solver

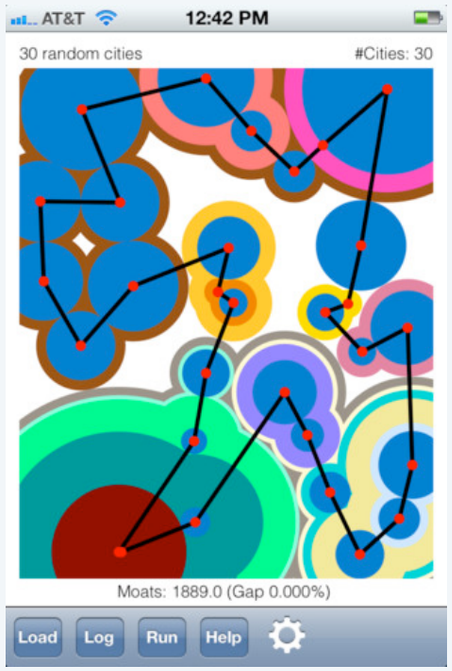
Concorde TSP solver. [Applegate–Bixby–Chvátal–Cook]

- Linear programming + branch-and-bound + polyhedral combinatorics.
- Greedy heuristics, including Lin–Kernighan.
- MST, Delaunay triangulations, fractional b-matchings, ...

Remarkable fact. Concorde has solved all 110 TSPLIB instances.

largest instance has 85,900 cities!





Thank You!