Hamilton Paths and Circuits

We have developed necessary and sufficient conditions for the existence of paths and circuits that contain every edge of a multigraph exactly once.

Can we do the same for simple paths and circuits that contain every vertex of the graph exactly once?

DEFINITION 1

A simple path in a graph G that passes through every vertex exactly once is called a Hamilton path, and

a simple circuit in a graph G that passes through every vertex exactly once is called a Hamilton circuit.

That is, the simple path $x_0, x_1, \ldots, x_{n-1}, x_n$ in the graph $G = (V, E)$ is a Hamilton path if $V = \{x_0, x_1, \ldots, x_{n-1}, x_n\}$ and $x_i \neq x_j$ for $0 \leq i < j \leq n$,

and the simple circuit $x_0, x_1, \ldots, x_{n-1}, x_n, x_0$ (with $n > 0$) is a Hamilton circuit if $x_0, x_1, \ldots, x_{n-1}, x_n$ is a Hamilton path.

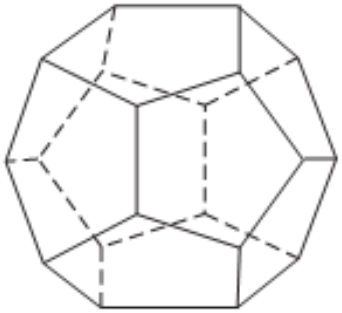
This terminology comes from a game, called the Icosian puzzle, invented in 1857 by the Irish mathematician Sir William Rowan Hamilton.

It consisted of a wooden dodecahedron [a polyhedron with 12 regular pentagons as faces, as shown bellow], with a peg at each vertex of the dodecahedron, and string.

The 20 vertices of the dodecahedron were labeled with different cities in the world.

The object of the puzzle was to start at a city and travel along the edges of the dodecahedron, visiting each of the other 19 cities exactly once, and end back at the first city.

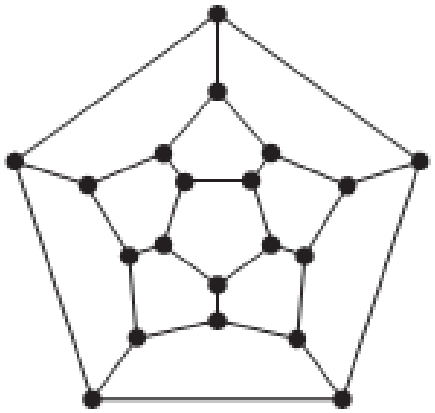
The circuit traveled was marked off using the string and pegs.



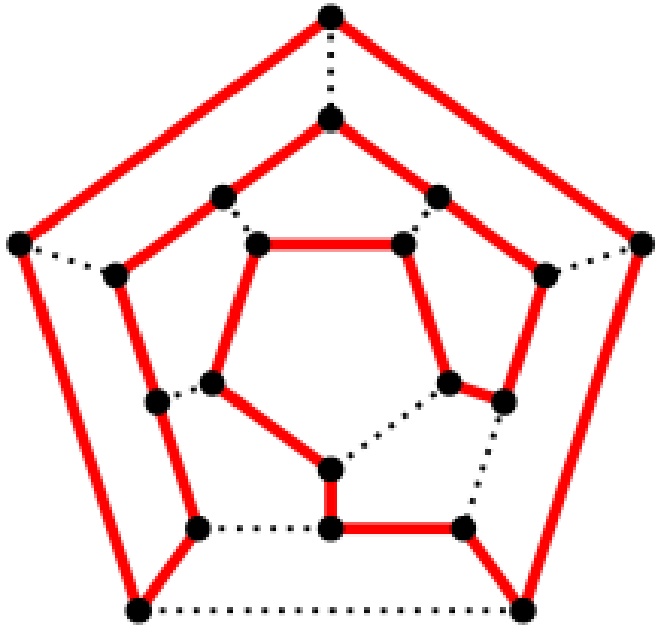
(a)

We will consider the equivalent question: Is there a circuit in the graph bellow that passes through each vertex exactly once?

This solves the puzzle because this graph is isomorphic to the graph consisting of the vertices and edges of the dodecahedron.



(b)



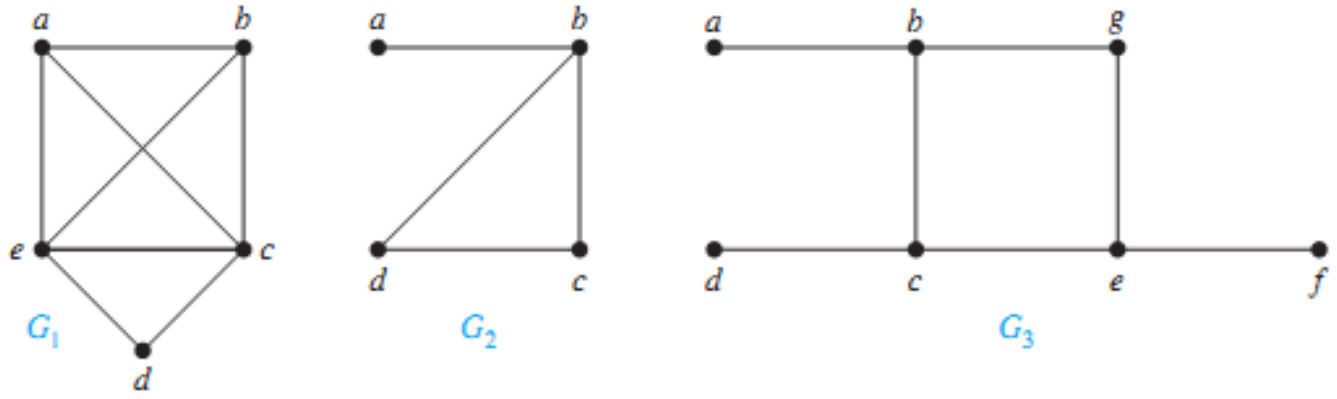
EXAMPLE 1

Which of the simple graphs bellow have a Hamilton circuit or, if not, a Hamilton path?

Solution: G_1 has a Hamilton circuit: *a, b, c, d, e, a*.

There is no Hamilton circuit in G_2 (this can be seen by noting that any circuit containing every vertex must contain the edge {*a, b*} twice), but G_2 does have a Hamilton path, namely, *a, b, c, d*.

G_3 has neither a Hamilton circuit nor a Hamilton path, because any path containing all vertices must contain one of the edges {*a, b*}, {*e, f* }, and {*c, d*} more than once.



EXAMPLE 2

Show that K_n has a Hamilton circuit whenever $n \geq 3$.

Solution: We can form a Hamilton circuit in K_n beginning at any vertex.

Such a circuit can be built by visiting vertices in any order we choose, as long as the path begins and ends at the same vertex and visits each other vertex exactly once.

This is possible because there are edges in K_n between any two vertices.

CONDITIONS FOR THE EXISTENCE OF HAMILTON CIRCUITS

Is there a simple way to determine whether a graph has a Hamilton circuit or path?

At first, it might seem that there should be an easy way to determine this, because there is a simple way to answer the similar question of whether a graph has an Euler circuit.

Surprisingly, there are no known simple necessary and sufficient criteria for the existence of Hamilton circuits.

However, many theorems are known that give sufficient conditions for the existence of Hamilton circuits.

Also, certain properties can be used to show that a graph has no Hamilton circuit.

1) A graph with a vertex of degree 1 cannot have a Hamilton circuit, because in a Hamilton circuit, each vertex is incident with 2 edges in the circuit.

2) If a vertex in the graph has degree 2, then both edges that are incident with this vertex must be part of any Hamilton circuit.

3) When a Hamilton circuit is being constructed and this circuit has passed through a vertex, then all remaining edges incident with this vertex, other than the 2 used in the circuit, can be removed from consideration.

4) A Hamilton circuit cannot contain a smaller circuit within it.

EXAMPLE 3

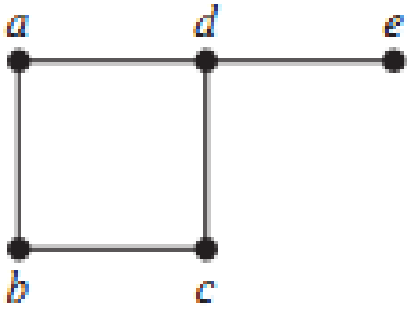
Show that neither graph displayed bellow has a Hamilton circuit.

Solution: There is no Hamilton circuit in *G* because *G* has a vertex of degree 1, namely, *e*.

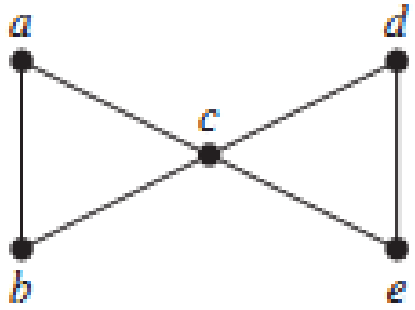
Now consider *H*.

Because the degrees of the vertices *a, b, d,* and *e* are all 2, every edge incident with these vertices must be part of any Hamilton circuit.

It is now easy to see that no Hamilton circuit can exist in *H*, for any Hamilton circuit would have to contain *4* edges incident with *c*, which is impossible.



G



H

Although no useful necessary and sufficient conditions for the existence of Hamilton circuits are known, quite a few sufficient conditions have been found.

Note that the more edges a graph has, the more likely it is to have a Hamilton circuit.

Furthermore, adding edges (but not vertices) to a graph with a Hamilton circuit produces a graph with the same Hamilton circuit.

So as we add edges to a graph, especially when we make sure to add edges to each vertex, we make it increasingly likely that a Hamilton circuit exists in this graph.

Consequently, we would expect there to be sufficient conditions for the existence of Hamilton circuits that depend on the degrees of vertices being sufficiently large.

We state 2 of the most important sufficient conditions here.

These conditions were found by Gabriel A. Dirac in 1952 and Øystein Ore in 1960.

DIRAC'S THEOREM

THEOREM 1 (DIRAC'S THEOREM) If G is a simple graph with n vertices with $n \geq 3$ such that the degree of every vertex in G is $\geq n/2$, then G has a Hamilton circuit.

ORE'S THEOREM

THEOREM 2 (ORE'S THEOREM):

If G is a simple graph with n vertices with $n \geq 3$ such that

$\deg(u) + \deg(v) \geq n$ for every pair of nonadjacent vertices u and v in G,

then G has a Hamilton circuit.

*Theorem 1 (*DIRAC'S THEOREM) If G is a simple graph with $n \geq 3$ and $\delta \geq n/2$, then G is hamiltonian.

Proof By contradiction. Suppose that the theorem is false, and let G be a maximal nonhamiltonian simple graph with $n \geq 3$ and $\delta \geq n/2$.

Since $n \geq 3$, G cannot be complete.

Let u and v be nonadjacent vertices in G.

By the choice of G, $G + \{u, v\}$ is hamiltonian.

Moreover, since G is nonhamiltonian, each Hamilton cycle of $G + \{u, v\}$ must contain the edge $\{u, v\}$.

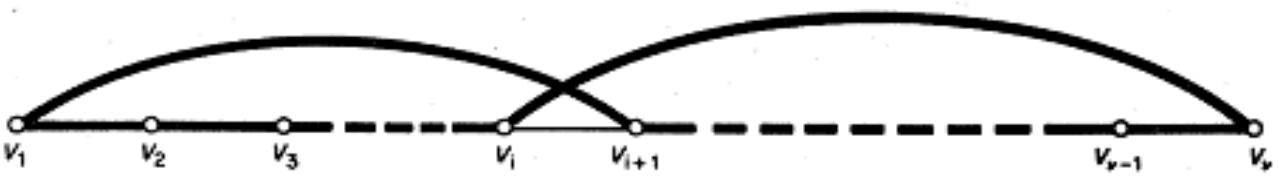
Thus there is a Hamilton path $v_1, v_2, \ldots v_n$ in G with origin $u = v_1$ and terminus $v = v_n$.

Set $S = \{v_i \mid \{u, v_{i+1}\} \in E\}$ and $T = \{v_i \mid \{v_i, v\} \in E\}$

Since $v_n \notin S \cup T$ we have $|S \cup T| < n$ (1)

Furthermore $|S \cap T| = \varnothing$ (2)

since if $S \cap T$ contained some vertex v_i, then G would have the Hamilton cycle $v_1, v_2, \ldots, v_i\, v_n, v_{n-1}, \ldots v_{i+1}, v_1$, contrary to assumption (see bellow).



Using (1) and (2) we obtain

$d(u) + d(v) = |S| + |T| = |S \cup T| + |S \cap T| < n$ (3)

But this contradicts the hypothesis that $\delta(G) \geq n/2$.

Both Ore's theorem and Dirac's theorem provide sufficient conditions for a connected simple graph to have a Hamilton circuit.

However, these theorems do not provide necessary conditions for the existence of a Hamilton circuit.

For example, the graph C_5 has a Hamilton circuit but does not satisfy the hypotheses of either Ore's theorem or Dirac's theorem.

The best algorithms known for finding a Hamilton circuit in a graph or determining that

no such circuit exists have exponential worst-case time complexity (in the number of vertices of the graph).

Finding an algorithm that solves this problem with polynomial worst-case time complexity would be a major accomplishment because it has been shown that this problem is NP-complete.

Consequently, the existence of such an algorithm would

imply that many other seemingly intractable problems could be solved using algorithms with polynomial worst-case time complexity.

Applications of Hamilton Circuits

Hamilton paths and circuits can be used to solve practical problems.

For example, many applications ask for a path or circuit that visits each road intersection in a city,

or each node in a communications network exactly once.

Finding a Hamilton path or circuit in the appropriate graph model can solve such problems.

The famous traveling salesperson problem or TSP (also known in older literature as the traveling salesman problem) asks for the shortest route a traveling salesperson should take to visit a set of cities.

This problem reduces to finding a Hamilton circuit in a complete graph such that the total weight of its edges is as small as possible.

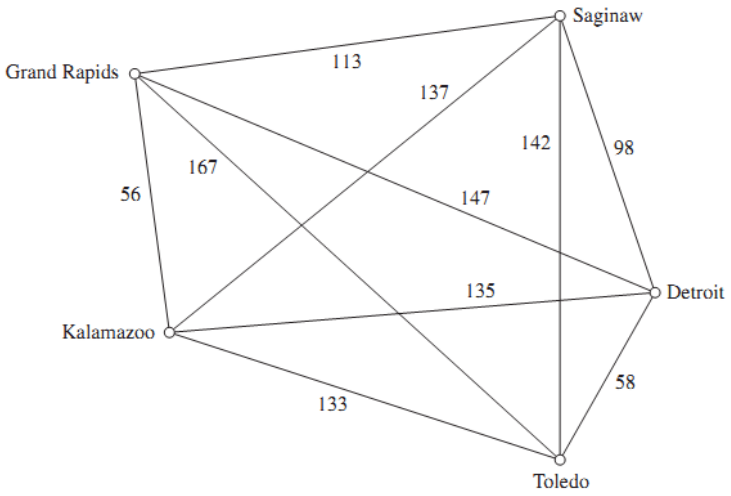
Example :The Traveling Salesperson Problem

Consider the following problem:

A traveling salesperson wants to visit each of *n* cities exactly once and return to his starting point.

For example, suppose that the salesperson wants to visit Detroit, Toledo, Saginaw, Grand Rapids, and Kalamazoo.

In which order should he visit these cities to travel the minimum total distance?



To solve this problem we can assume the salesperson starts in Detroit (because this must be part of the circuit) and examine all possible ways for him to visit the other four cities and then return to Detroit (starting elsewhere will produce the same circuits).

There are a total of 4*3*2*1 = 24 such circuits, but because we travel the same distance when we travel a circuit in reverse order, we need only consider 12 different circuits to find the minimum total distance he must travel.

We list these 12 different circuits and the total distance traveled for each circuit.

As can be seen from the list, the minimum total distance of 458 miles is traveled using the circuit Detroit–Toledo–Kalamazoo–Grand Rapids–Saginaw–Detroit (or its reverse).

Route	*Total Distance (miles)*
Detroit–Toledo–Grand Rapids–Saginaw–Kalamazoo–Detroit	610
Detroit–Toledo–Grand Rapids–Kalamazoo–Saginaw–Detroit	516
Detroit–Toledo–Kalamazoo–Saginaw–Grand Rapids–Detroit	588
Detroit–Toledo–Kalamazoo–Grand Rapids–Saginaw–Detroit	458
Detroit–Toledo–Saginaw–Kalamazoo–Grand Rapids–Detroit	540
Detroit–Toledo–Saginaw–Grand Rapids–Kalamazoo–Detroit	504
Detroit–Saginaw–Toledo–Grand Rapids–Kalamazoo–Detroit	598
Detroit–Saginaw–Toledo–Kalamazoo–Grand Rapids–Detroit	576
Detroit–Saginaw–Kalamazoo–Toledo–Grand Rapids–Detroit	682
Detroit–Saginaw–Grand Rapids–Toledo–Kalamazoo–Detroit	646
Detroit–Grand Rapids–Saginaw–Toledo–Kalamazoo–Detroit	670
Detroit–Grand Rapids–Toledo–Saginaw–Kalamazoo–Detroit	728

General case

The traveling salesperson problem asks for the circuit of minimum total weight in a weighted, complete, undirected graph that visits each vertex exactly once and returns to its starting point.

This is equivalent to asking for a Hamilton circuit with minimum total weight in the complete graph, because each vertex is visited exactly once in the circuit.

The most straightforward way to solve an instance of the traveling salesperson problem is to examine all possible Hamilton circuits and select one of minimum total length.

How many circuits do we have to examine to solve the problem if there are *n* vertices in the graph?

Once a starting point is chosen, there are $(n - 1)!$ different Hamilton circuits to examine, because there are $n - 1$ choices for the second vertex, $n - 2$ choices for the third vertex, and so on.

Because a Hamilton circuit can be traveled in reverse order, we need only examine $(n - 1)!/2$ circuits to find our answer.

Note that $(n - 1)!/2$ grows extremely rapidly.

Trying to solve a traveling salesperson problem in this way when there are only a few dozen vertices is impractical.

For example, with 25 vertices, a total of $24!/2$ (approximately 3.1×10^{23}) different Hamilton circuits would have to be considered.

If it took just 1 nanosecond (10^{-9} second) to examine each Hamilton circuit, a total of approximately 10 million years would be required to find a minimum-length Hamilton circuit in this graph by exhaustive search techniques.

Because the traveling salesperson problem has both practical and theoretical importance, a great deal of effort has been devoted to devising efficient algorithms that solve it.

However, no algorithm with polynomial worst-case time complexity is known for solving this problem.

Furthermore, if a polynomial worst-case time complexity algorithm were discovered for the traveling salesperson problem, many other difficult problems would also be solvable using polynomial worst-case time complexity algorithms (such as determining whether a proposition in n variables is a tautology).

This follows from the theory of *NP*- completeness.

A practical approach to the traveling salesperson problem when there are many vertices to visit is to use an approximation algorithm.

These are algorithms that do not necessarily produce the exact solution to the problem but instead are guaranteed to produce a solution that is close to an exact solution.

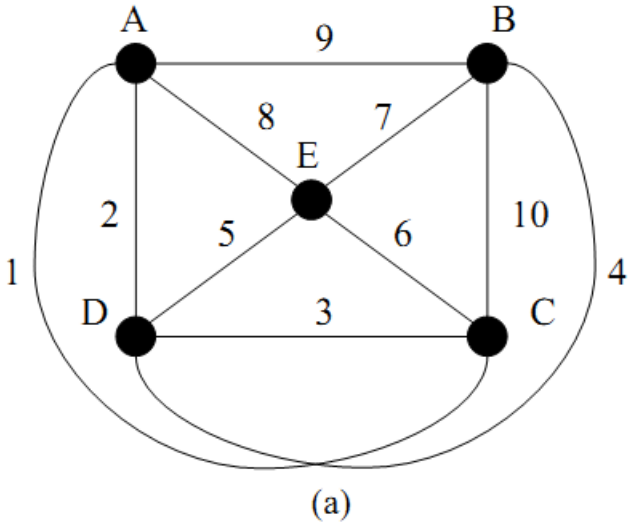
In practice, algorithms have been developed that can solve traveling salesperson problems with as many as 1000 vertices within 2% of an exact solution using only a few minutes of computer time.

CHEAPEST LINK ALGORITHM (CLA)

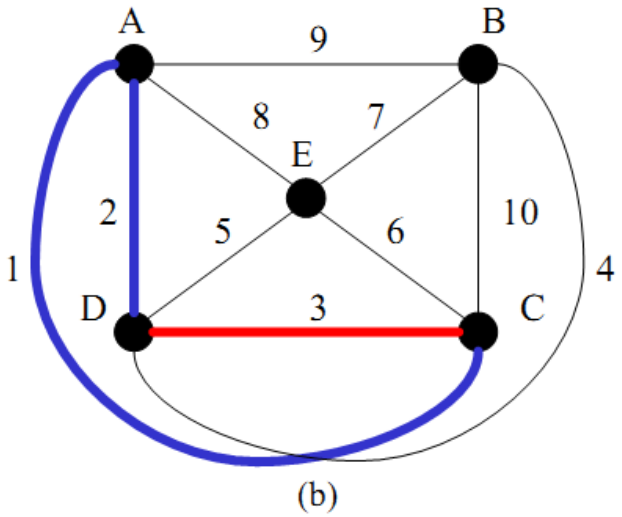
1. Choose the edge with the smallest weight (the "cheapest" edge), randomly breaking ties
2. Keep choosing the "cheapest" edge unless it
    (a) closes a smaller circuit OR
    b) results in 3 selected edges coming out of a single vertex

Continue until the Hamilton Circuit is complete .

CHEAPEST LINK ALGORITHM (CLA)



(a)



(b)

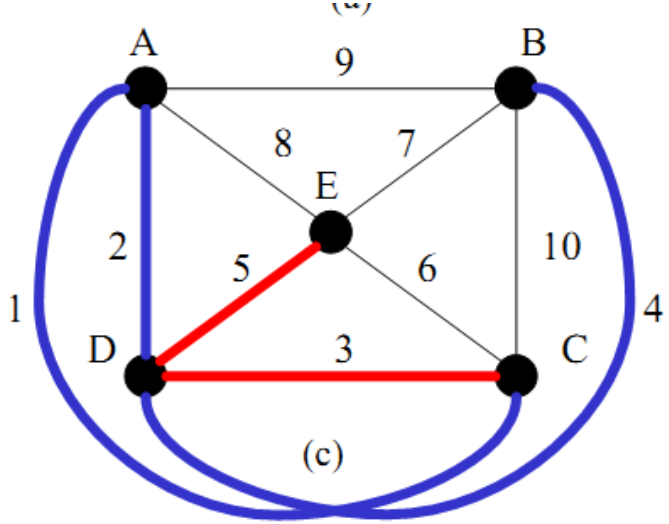
Example:

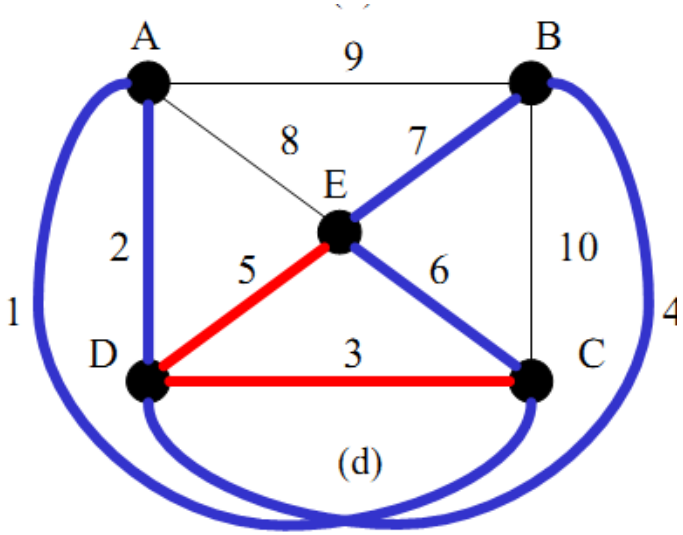
Apply this algorithm on a graph K_5 :

1)We choose AC (1) (min weight).

2)Then, since no restriction will be validated we add AD (2).

3)Now if we try to add DC (3)we get a cycle ACDA so we do not add it.





Instead we take the next lower weighted edge DB (4) and add it.

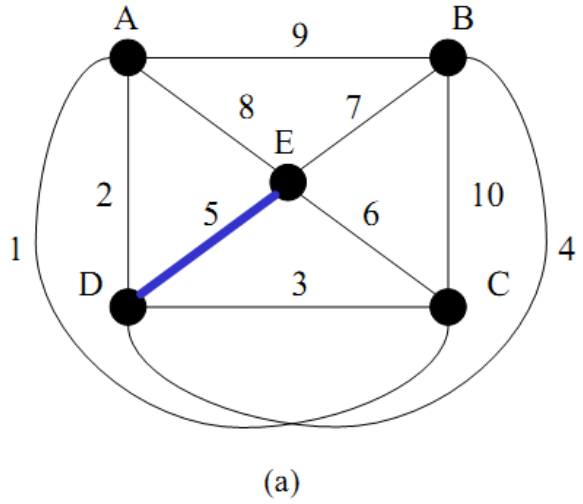
If we add ED (5) node D will have degree 3, so we do not use it.

Instead we add EC(6) and EB(7) successively so we get a Hamiltonian cycle ADBECA with total cost 20

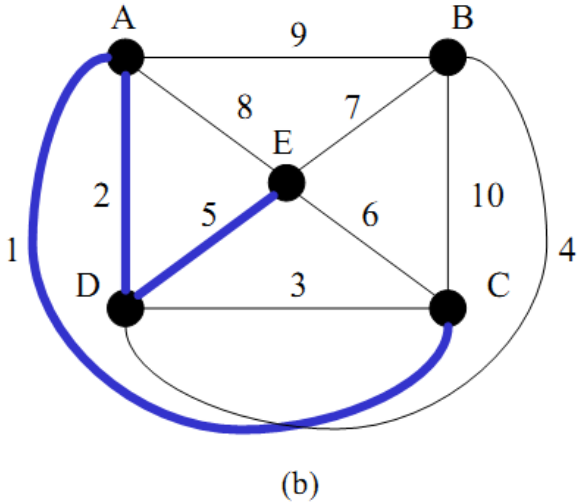
NEAREST NEIGHBOR ALGORITHM (NNA)

1. Start at a vertex (think of it as your Home city)
2. Travel to the vertex that you haven't been to yet whose path has the smallest weight. If there is a tie, pick randomly.
3. Continue until you travel to all vertices
4. Travel back to your starting vertex

NEAREST NEIGHBOR ALGORITHM (NNA)



(a)



(b)

Example Apply NNA to a graph K_5

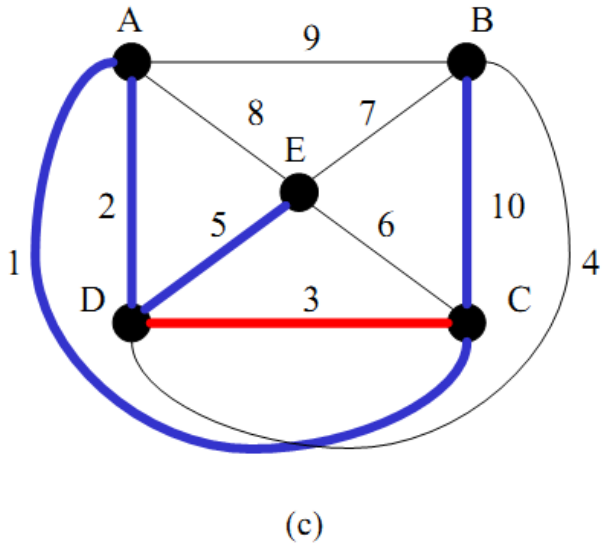
We choose randomly E as the initial vertex.

We choose the edge with the lower cost starting from E which is ED (5).

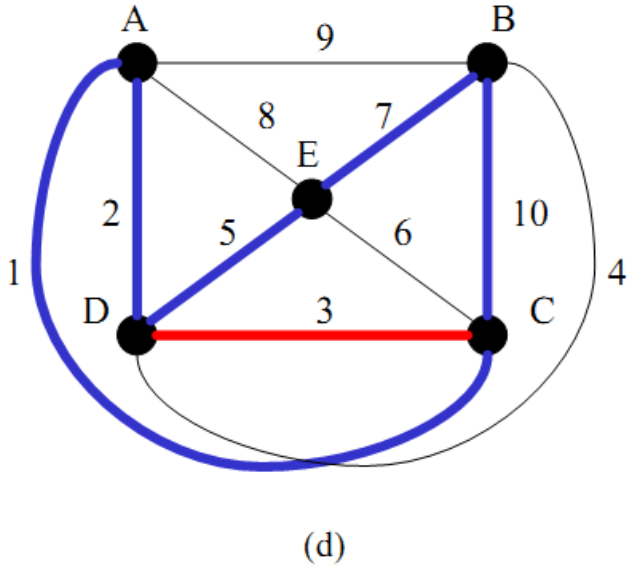
Same way we add DA(2) and AC(1).

Here we can not use the edge with the lower cost CD(3) since D has already been visited

NEAREST NEIGHBOR ALGORITHM (NNA)



(c)



(d)

We travel to B through CB(10) and finally reach the circuit EDACBE with total cost 25 (worse than the previous)

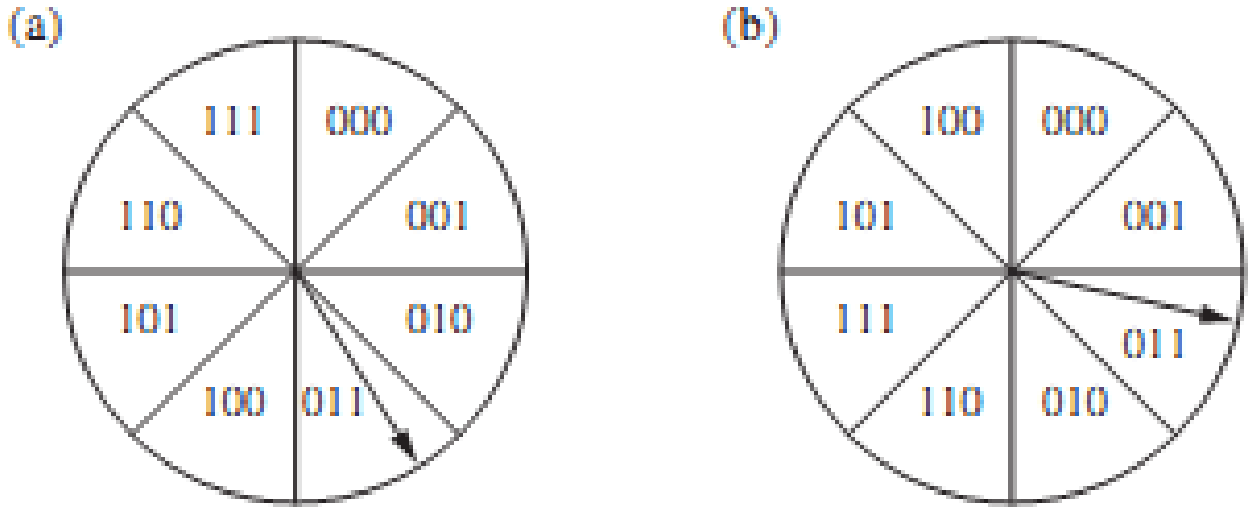
Example 4. Gray codes

We now describe a less obvious application of Hamilton circuits to coding.

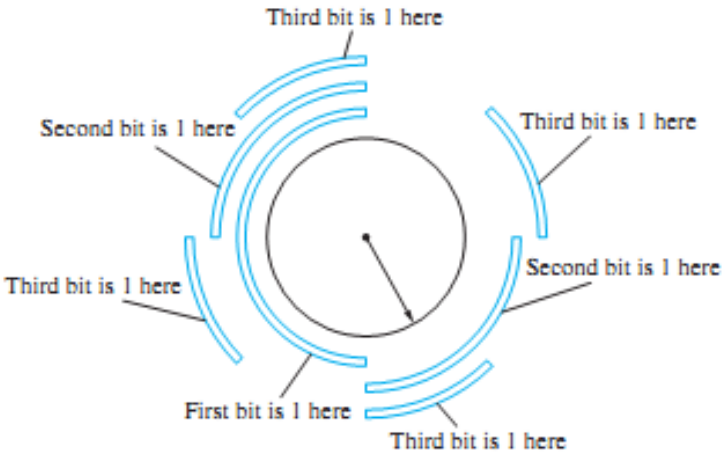
The position of a rotating pointer can be represented in digital form.

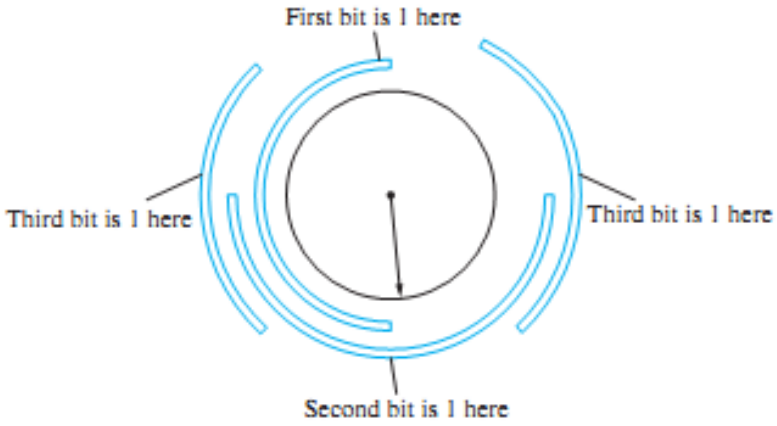
One way to do this is to split the circle into 2^n arcs of equal length and to assign a bit string of length n to each arc.

2 ways to do this using bit strings of length 3 are shown bellow.



The digital representation of the position of the pointer can be determined using a set of *n* contacts.
Each contact is used to read 1 bit in the digital representation of the position.





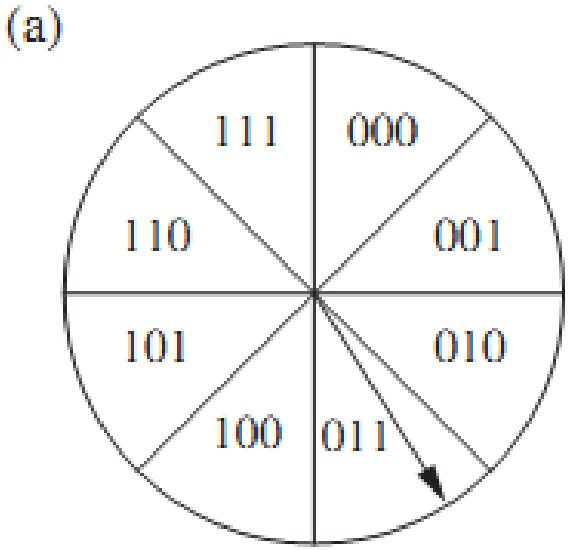
EXAMPLE 4 Gray Codes

When the pointer is near the boundary of 2 arcs, a mistake may be made in reading its position.

This may result in a major error in the bit string read.

For instance, in the coding scheme bellow, if a small error is made in determining the position of the pointer, the bit string 100 is read instead of 011.

All three bits are incorrect!



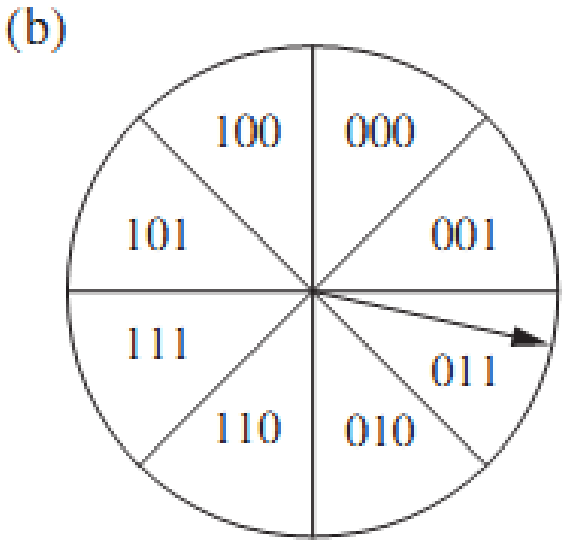
To minimize the effect of an error in determining the position of the pointer,

the assignment of the bit strings to the 2^n arcs

- should be made so that only 1 bit is different in the bit strings represented by adjacent arcs.

This is exactly the situation in the coding scheme bellow.

- An error in determining the position of the pointer gives the bit string 010 instead of 011. Only 1 bit is wrong.



The assignment is a Gray code.

A Gray code is a labeling of the arcs of the circle such that adjacent arcs are labeled with bit strings that differ in exactly 1 bit.

We can find a Gray code by listing all bit strings of length *n* in such a way that each string differs in exactly *1* position from the preceding bit string, and the last string differs from the first in exactly one position.

We can model this problem using the *n*-cube Q_n.

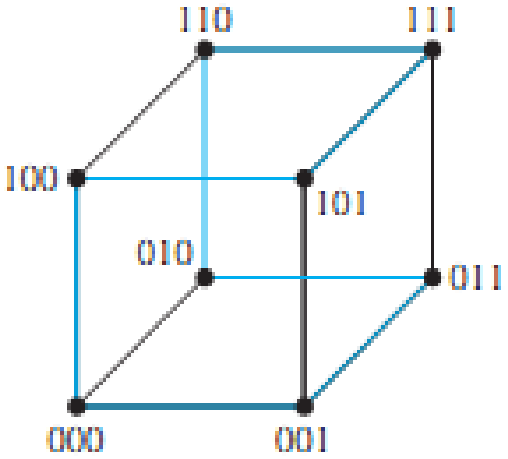
- What is needed to solve this problem is a Hamilton circuit in Q_n.

Such Hamilton circuits are easily found .

For instance, a Hamilton circuit for Q_3 is displayed bellow.

The sequence of bit strings differing in exactly one bit produced by this Hamilton circuit is 000, 001, 011, 010, 110, 111, 101, 100.

Gray codes are named after Frank Gray, who invented them in the 1940s at AT&T Bell Laboratories to minimize the effect of errors in transmitting digital signals.



A Hamilton Circuit for Q_3.