

**Title: The Chinese Postman Problem (CPP) and It’s Solution**

**Author Name:** Mamo Tadiyos Hailemichael

**School:** Computer Science and Engineering

**Major:** Computer Science and Technology

**Teachers Name:** Professor Gao Hui

**Course Title:** Discrete Mathematics

01 **​**July, **​ ​**2020

Table of Contents

Description Page No.

0 **Abstract** -------------------------------------------------------------------------------------------------------------------------3

1  **Curriculum Design Tasks and Requirements:** -------------------------------------------------------------------------4

1.1 Description to Chinese Postman Problem (CPP) ---------------------------------------------------------4

1.2 Pre-request Concepts and Terminologies of Graph----------------------------------------------------- 6

1.2.1 Traversable graph ----------------------------------------------------------------------------------- 6

1.2.2 Pairing Odd Vertices --------------------------------------------------------------------------------8

**2 Fundamentals:** ----------------------------------------------------------------------------------------------------------------9

2.1 Definitions and Outline of the Algorithm-------------------------------------------------------------------9

**3 Theoretical Design (and Implementation):** --------------------------------------------------------------------------10

3.1 Solution for Chinese Postman Problem------------------------------------------------------------------10

3.1.1 Finding an Optimal Route-------------------------------------------------------------------------11

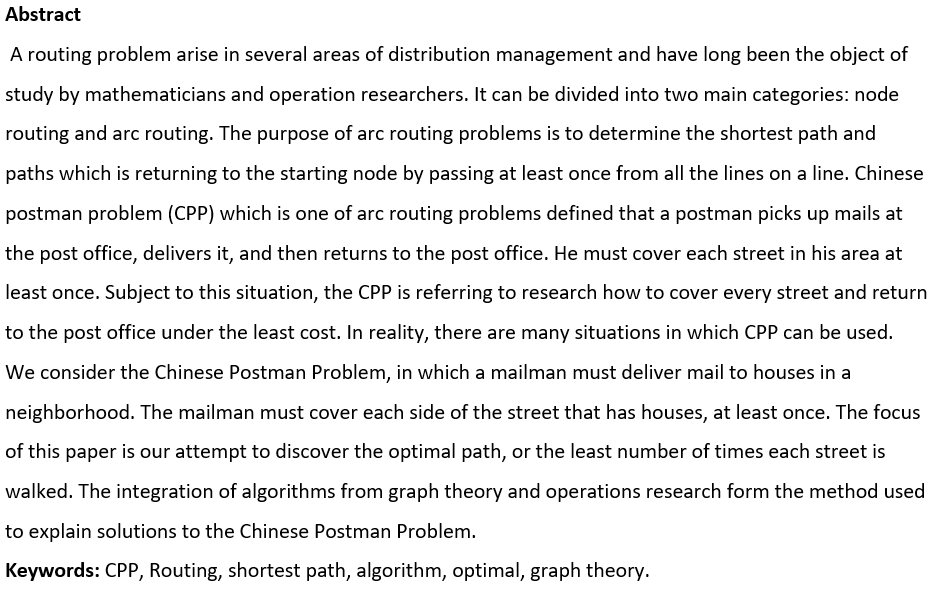
3.2 Implementation of CPP Algorithm -------------------------------------------------------------------------12

**4 Conclusions Analysis** -------------------------------------------------------------------------------------------------------16

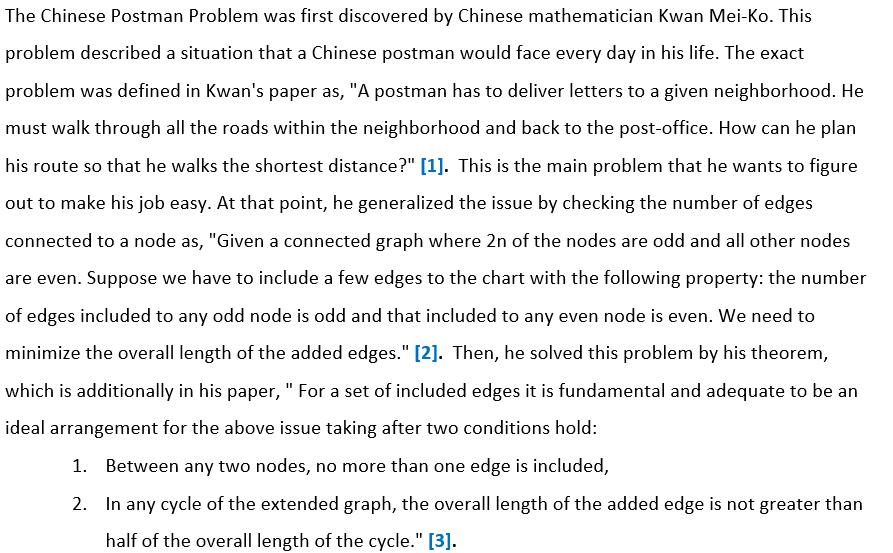
**5 Harvest, experience and suggestions** ---------------------------------------------------------------------------------17

**6 Acknowledgements** --------------------------------------------------------------------------------------------------------18

**7 Reference** ---------------------------------------------------------------------------------------------------------------------18



1. **Curriculum Design Tasks and Requirements:**
   1. **Description to Chinese Postman Problem (CPP)**



The Chinese postman who wishes to travel along each street to deliver letters. The Chinese Postman Problem (CPP) is curiously since it has numerous applications, could be a simply-stated issue, but for which there's no straightforward calculation. There are numerous varieties to the CPP, most strikingly whether the streets are one-way (typically the Directed CPP or DPP) and whether the postman has got to return back to where they begun (closed or open CPP).This paper is concerned specifically with the directed CPP, and provides algorithms for both closed and open solutions.

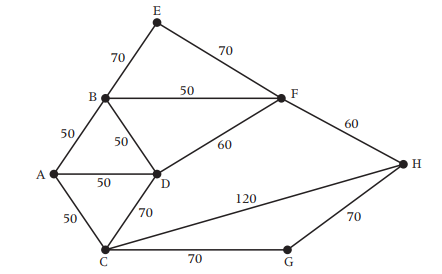
The CPP is one of the foremost interesting problems in Operations Research (OR). It determines its title from the fact that an early paper discussing this problem appeared in the journal Chinese Mathematics **[4]**. The CPP has a simple goal, to transverse each edge of a graph at least once with a minimum of backtracking. The great Swiss mathematician Leonhard Euler first examined this problem in the 18th century. Euler attempted to discover a way in which a parade procession may cross all 7 bridges   
exactly once in the city of Konigsberg, Prussia **[5]**. Euler demonstrated in 1736 that no solution to the onigsberg routing issue exists. He also determined a few common results that give the inspiration for the solution to the CPP. These results will be introduced in detail below.

The Chinese postman problem, also known as the route inspection problem, is to discover a circuit in an associated undirected graph that visits each edge at least once. Also, an adjusted arrangement minimizes the full fetched cost of the circuit. In case all vertices of the graph have even degree, at that point the ideal arrangement is the Euler circuit of the graph. In this case, the fetched cost of the arrangement is the sum of whole edge weights in the graph.

Within the case where the graph has a few odd edges, the arrangement is more complicated. To begin with identifying all nodes with odd degree. Following, match up the odd nodes such that the entire sum of the weights of the least paths between pairs is a least. At that point, take all the edges from the paths in the past step, and copy them. This guarantees that all nodes presently have even degree, whereas including the least conceivable weight to the graph. At last, the arrangement is the Euler circuit of the altered graph, and the fetched cost is the entire sum of whole edge weights in the original graph added with the entire sum of the weights of the copied edges.

The three primary steps (discover odd nodes, discover least paths, and discover Euler circuit) can all be figured out in polynomial time, so this problem can be illuminated in polynomial time.

Within the following example a postman should begin at A, walk along all 13 roads and return to A. The numbers on every edge denotes the length, in meters, of all street. The problem is to discover a path that uses entire edges of a graph with least length.

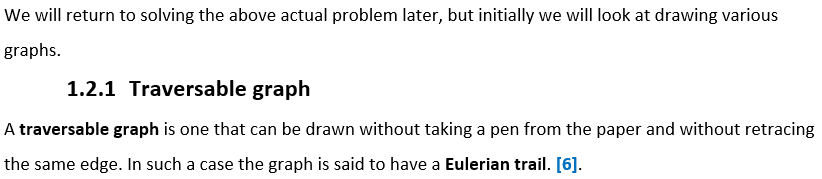


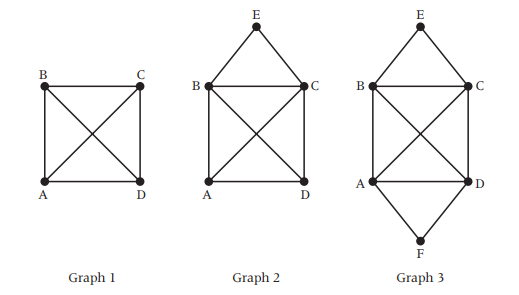
How seem a postman visit each letter on the graph in the shortest conceivable time?

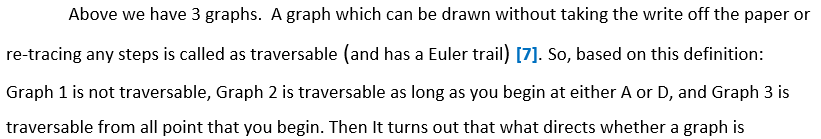
Tackling this requires using a branch of mathematical science called **graph theory**, made by Leonard Euler.  This arithmetic looks to decrease problems to network graphs like that shown above.

Before we can solve the above graph problem, we ought to get into a few terminologies on next page:

* 1. **Pre-request Concepts and Terminologies of Graph**





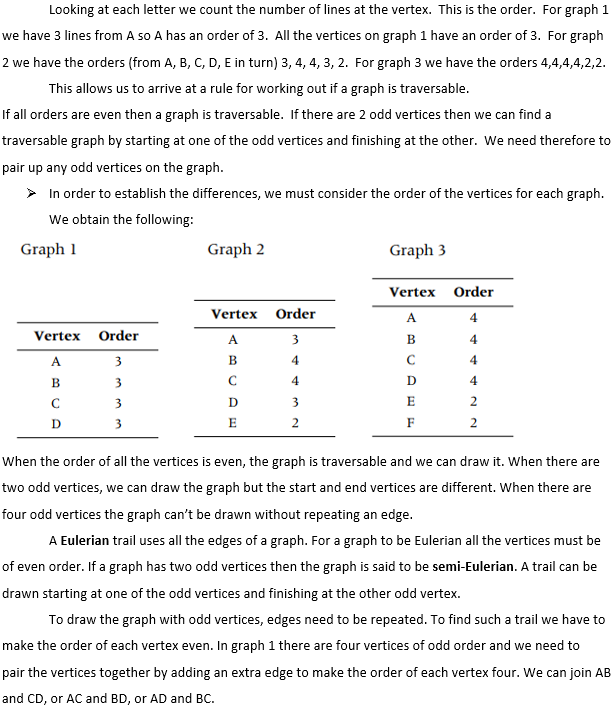


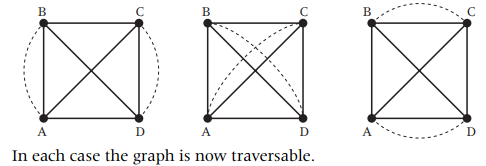
traversable or not is the arrange of their vertices.

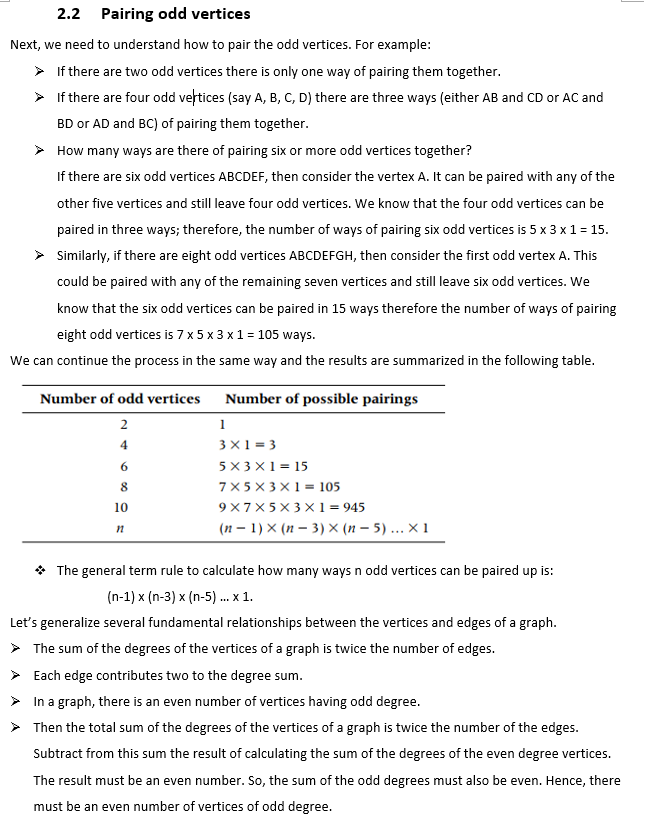
Generally, in case we attempt drawing the three graphs shown above, we find:

* It is inconceivable to draw Graph1 without either taking the write off the paper or retracing an edge.
* We can draw Graph 2, but as it were by beginning at either A or D – in all case the route will conclude at the other vertex of D or A.
* Graph 3 can be drawn in any case of the beginning position and you’ll continuously return to the beginning vertex.

What is the difference between these three graphs?



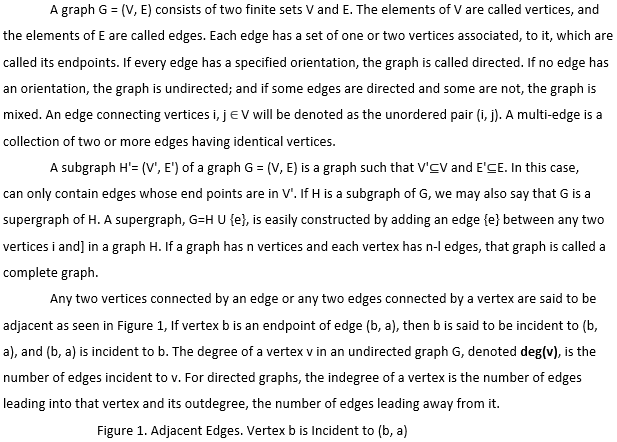


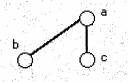




* A connected undirected graph is a Euler Graph thus processes a Euler Tour if and only if all vertices of the graph have even degree. **[8]**.

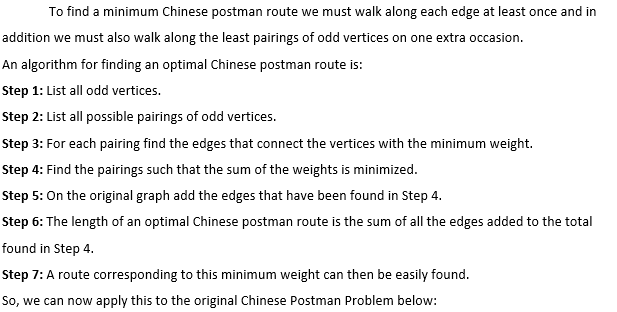
1. **Fundamentals:**
   1. **Definitions and Outline of the Algorithm**

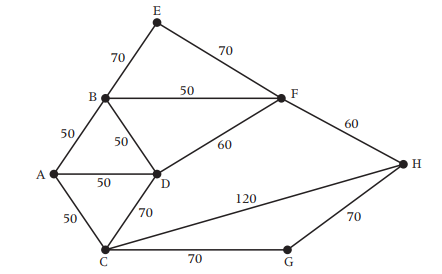




The characteristics of graphs we are interested in are identified by examination. Keeping in mind that my graph will be finite, the first characteristic we find is the graph could consist of nothing but even degree vertices. The degree of a vertex is the number of edges incident to that vertex. Euler's theorem states that if all vertices have even degree then there exists a Euler Tour, a tour which transverses every edge of a graph exactly once, and the optimal solution is ensured. **[9]**.

1. **Theoretical Design (and Implementation):**
   1. **Solution for Chinese Postman Problem**

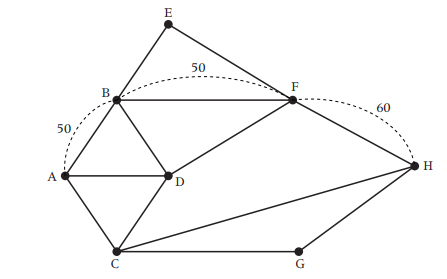


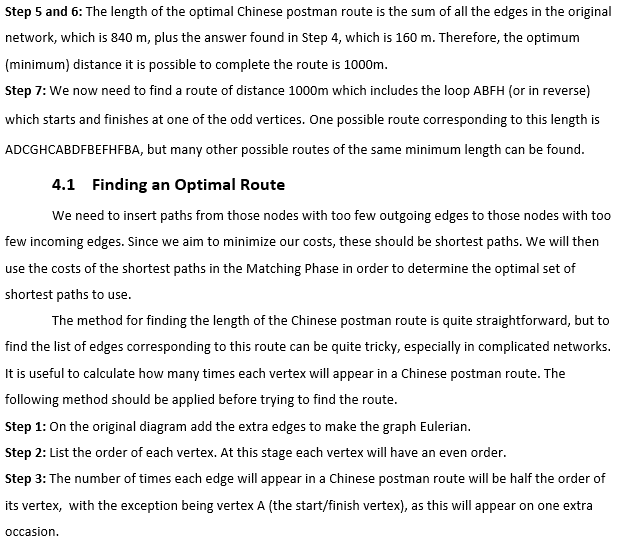


Let’s apply the above Chinese Postman Algorithm to this graph as follow:

**Step 1:** We are able see that the only odd vertices are A and H.

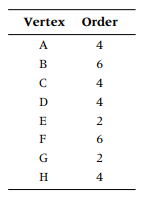
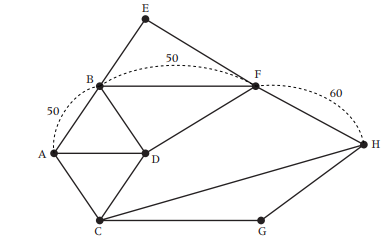
**Step 2:** There’s only one way of matching these odd vertices, to be specific AH.  
**Step 3:** The least possible way of joining A to H is using the path AB, BF, FH (ABFH), a total length of: 50 + 50 + 60 = **160**. This is shown below:  
**Step 4:** Draw these edges onto the above original network.







Referring to the diagram below, the orders of the vertices are as follows:

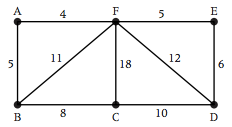
This indicates that the number of times each vertex will appear in the Chinese postman route is:

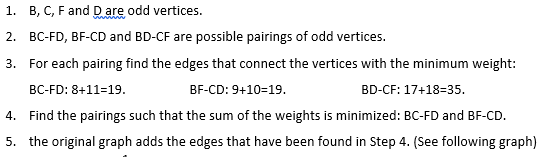
The number of vertices in the optimal Chinese postman route is:

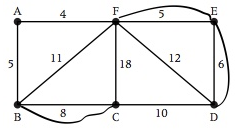
3 + 3 + 2 + 2 + 1 + 3 + 1 + 2 = **17**.  
They may be in a different order than in the example above but they must have the number of vertices as indicated in the table

A = 2 + 1 = 3 B = 3  
C = 2 D = 2  
E = 1 F = 3  
G = 1 H = 2

Let’s again apply the above algorithm for the following example of graph to find a minimum Chinese postman route.

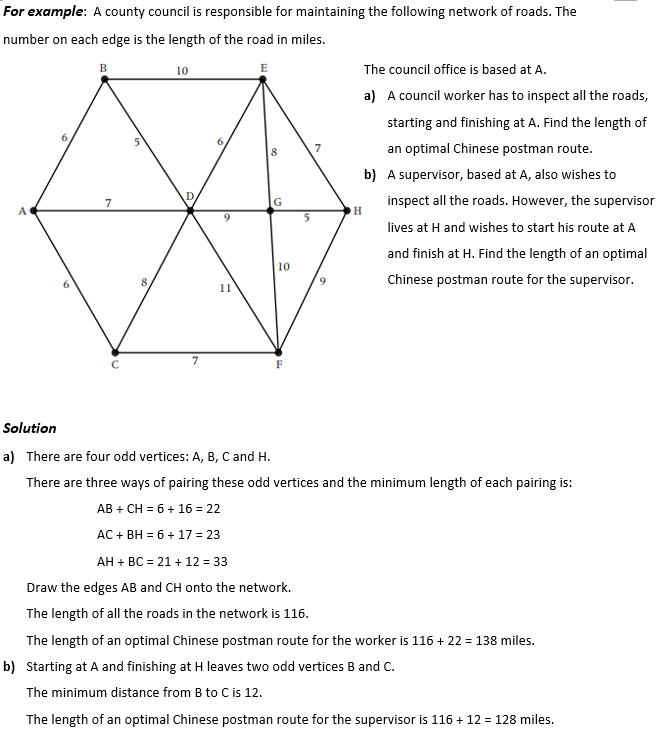




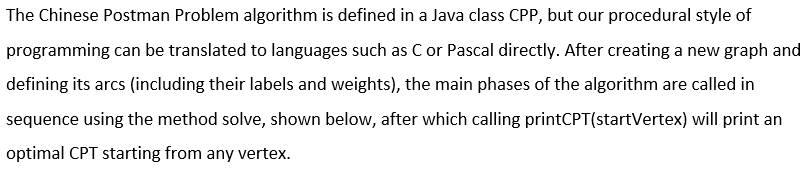


1. Total weight should be 79+19=**98**.
2. A possible route of this weight can be **AFEDFEDCFBCBA**

Let’s see our final example of real-world application that how a supervisor and a council worker maintain the following road networks.



* 1. **Implementation of CPP Algorithm**

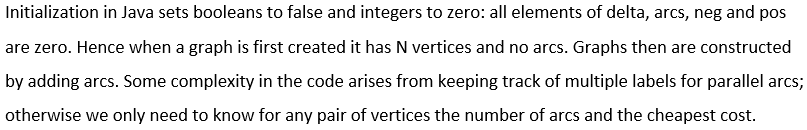


|  |
| --- |
| import java.io. \*;  import java.util.\*;  public class CPP  {  int N; // number of vertices  int delta []; // deltas of vertices  int neg [], pos []; // unbalanced vertices  int arcs [] []; // adjacency matrix, counts arcs between vertices  Vector label [] []; // vectors of labels of arcs (for each vertex pair)  int f [] []; // repeated arcs in CPT  float c [] []; // costs of cheapest arcs or paths  String cheapestLabel [] []; // labels of cheapest arcs  boolean defined [] []; // to check path cost is defined between vertices or not  int path [] []; //  float basicCost; // sum of the cost that traverse each arc once  void solve ()  {  leastCostPaths ();  checkValid ();  findUnbalanced ();  findFeasible ();  while (improvements ());  }  .  .  .  // Other definitions and declarations can be described below  .  .  .  } |



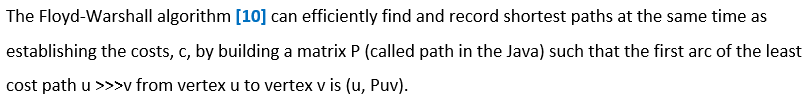
the algorithm. To avoid the complexity of changing the size of vectors and matrices dynamically, a CPP graph is instantiated with a fixed size:

|  |
| --- |
| // allocate array memory, and instantiate graph object  CPP (int vertices)  {  if ((N = vertices) <= 0) throw new Error ("Graph is empty");  delta = new int[N];  defined = new boolean[N][N];  label = new Vector[N][N];  c = new float[N][N];  f = new int[N][N];  arcs = new int[N][N];  cheapestLabel = new String[N][N];  path = new int[N][N];  basicCost = 0;  } |



|  |
| --- |
| CPP addArc (String lab, int u, int v, float cost)  {  if (! defined[u][v]) label[u][v] = new Vector ();  label[u][v]. addElement(lab);  basicCost += cost;  if (! defined[u][v] || c[u][v] > cost)  {  c[u][v] = cost;  cheapestLabel[u][v] = lab;  defined[u][v] = true;  path[u][v] = v;  }  arcs[u][v] ++;  delta[u]++;  delta[v]--;  return this;  }  // The final line, return this, allows addArc calls to be strung together conveniently. |

**Find shortest paths and costs**

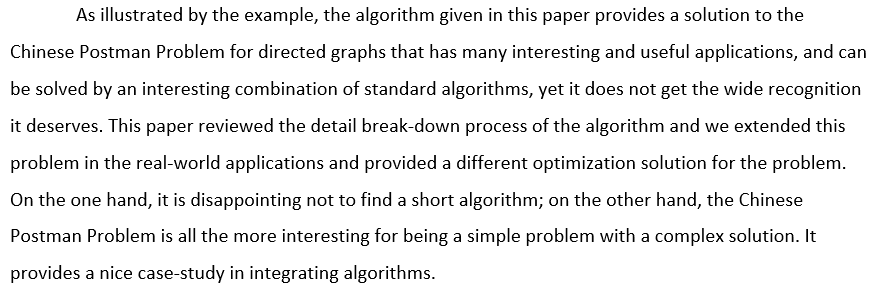


Subsequent arcs along the path are found similarly; if (u, Puv) is an arc to vertex w ≠v then (u, Pwv) is the next arc to take towards v, and so on.

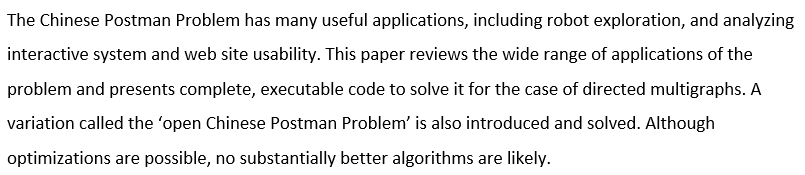
The standard algorithm is modified to terminate if any negative cycle is found: because it terminates before possibly finding further cycles through the same vertex, the cycle is correctly defined by recording a single value in path

|  |
| --- |
| void leastCostPaths ()  {  for (int k = 0; k < N; k++)  for (int i = 0; i < N; i++)  if(defined[i][k])  for (int j = 0; j < N; j++)  if(defined[k][j] && (! defined[i][j] || c[i][j] > c[i][k] +c[k][j]))  {  path[i][j] = path[i][k];  c[i][j] = c[i][k] +c[k][j];  defined[i][j] = true;  if (i == j && c[i][j] < 0) return; //stop on negative cycle  }  } |

1. **Conclusion Analysis**

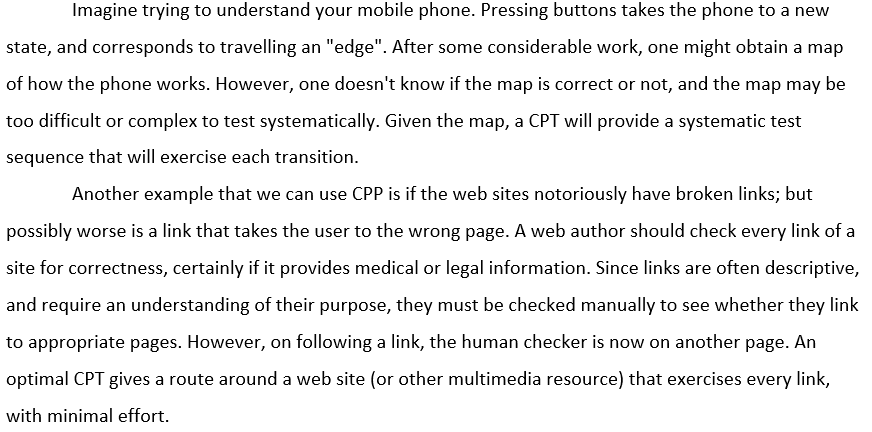


**5. Harvest, experience and suggestions**



CPP may be used in several areas, the main real-world application areas are:

* Mail delivery
* Garbage collection
* Road gritting
* Highways streets cleaning, ice controls and also snow removal operations
* School bus and police patrol vehicles routing
* Water and newspaper distribution
* Effective web site determination
* Network algorithm checking and transmission line inspections
* For well-designed web sites, to check every link explicitly



**References**

**[1]** Brucker, P. 1981. The Chinese Postman Problem for Mixed Graphs. In Graph Theoretic Concepts in Computer Science. H. Noltemeier (ed.). Springer, Berlin, 254-366

**[2]** Edmonds, J. 1965a. The Chinese Postman’s Problem. ORSA Bull. 13,73

**[3]** Win, Z. 1987. On the Windy Postman Problem on Eulerian Graphs. Math. Prog. 44, 97-112  
**[4]** Picard, J.-C., and H. D. Ratliff. 1975. Minimum Cuts and Related Problems. Networks 5, 357-370  
**[5]** Kaufmann, C. H. and G. J. Koehler. 1979. The Mixed Postman Problem. Discr. Appl. Math. 1, 89-103   
**[6]** Chrirsofides, N. 1975. Graph Theory. An Algorithm Approach. Academic Press, London   
**[7]** Frederickson 1978. Approximation Algorithms for Some Routing Problems. SIAM J. Comp. 7, 78-93

**[8]** Alfa, A. S., and D. Q. Liu. 1988. Postman Routing Problem in a Hierarchical Network; 127-138

**[9]** Harary, F. 1969. Graph Theory. Addison-Wesley, Reading, Mass

**[10]** Guan, M. 1962. Graph Programming Using Odd and Even Points. Chinese Math. 1, 273-277