

# Approximation Algorithms

Advanced Algorithms (CSE 794)

Instructor: Tamal K. Dey

Scribe: Sreeram Potluri \*

May 26, 2009

## 1 Introduction

While polynomial time algorithms to solve NP-Complete problems remain undiscovered, many such problems are too important to ignore. The following techniques can be used to work around such problems in real world.

- Exponential Algorithms - When input size is small, using exponential algorithms may not be costly.
- Approximation - Finding a near-optimal solution might be much simpler.
- Restriction - Restricting the structure of the input might make faster algorithms possible.
- Parameterization - Fixing some input parameters may help solve the problem faster.
- Heuristic - An algorithm that works reasonably well but no proof exists that it is fast and correct.

This document covers polynomial-time approximation algorithms for two well-known optimization problems that are NP-Complete: the Vertex-cover Problem and the Travelling Salesperson Problem

## 2 Performance Ratios for Approximation Algorithms

### 2.1 Approximation Ratio

For an optimization problem where the goal is to optimize the cost, we say that an algorithm has an approximation ratio  $\rho(n)$  if, for any input size  $n$ , the cost  $C$  of the solution produced by the algorithm is within a factor of  $\rho(n)$  of the cost  $C^*$  of the optimal solution. We can also call such an algorithm a  $\rho(n)$  approximation algorithm.

$$\max\left(\frac{C}{C^*}, \frac{C^*}{C}\right) \leq \rho(n) \quad (1)$$

For a maximization problem,  $0 < C \leq C^*$ , and the ratio  $C^*/C$  gives the factor by which optimal solution is larger than the approximate solution. Similarly, for a minimization problem  $0 < C^* \leq C$ , and the ratio  $C/C^*$  gives the factor by which the approximate solution is larger than the optimal solution. An algorithm is said to give an optimal solution if  $C = C^*$  and  $\rho(n) = 1$ .

---

\*Department of Computer and Information Sciences, The Ohio State University, Columbus, Ohio, USA.

While it is good to have a constant approximation factor, for some problems, the best known algorithms have approximation factors that grow as a function of the input size  $n$ .

## 2.2 Approximation Scheme

Often, there exists a trade-off between the time complexity of an algorithm and its approximation ratio. The smaller the approximation factor, the greater can be the time complexity. This trade-off is characterized using an Approximation Scheme. An Approximation Scheme for an optimization problem is an approximation algorithm that takes as input not only an instance of the problem, but also a value  $\epsilon > 0$  and can produce a solution that is within a factor  $\epsilon$  from the optimal solution. We call this a polynomial-time approximation scheme if the algorithm runs in polynomial time for the given  $\epsilon$ .

In some cases, the lesser the value of  $\epsilon$ , the greater will be the time complexity of the algorithm. Eg:  $O(n^{\frac{1}{\epsilon}})$ . In the ideal case, a constant factor decrease in  $\epsilon$  should cause only a corresponding constant factor increase in the time complexity. We call such a scheme a fully polynomial-time approximation scheme. Eg:  $O(\frac{1}{\epsilon}n^2)$ .

## 3 The Vertex-Cover Problem

A Vertex Cover of an undirected graph  $G = (V, E)$  is a subset  $V' \subseteq V$  such that if  $(u, v)$  is an edge of  $G$ , then either  $u \in V'$  or  $v \in V'$  (or both). The size of the vertex cover is the number of vertices in it. The Vertex-Cover problem is to find such a subset of vertices that is minimal. This problem has been proven to be an NP-Complete problem.

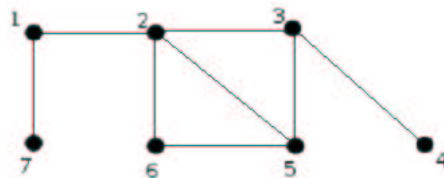


Figure 1: The minimal vertex cover of this graph is (1, 6, 3)

However, a sub-optimal solution which is no more than twice the optimal solution can be found in polynomial time. We first present the algorithm below and then go on to prove that it is indeed a 2-approximation algorithm.

APPROX-VERTEX-COVER( $G$ )

1.  $C \leftarrow \phi$
2.  $E' \leftarrow E[G]$
3. **while**  $E' \neq \phi$
4.     **do** Let  $(u, v)$  be an arbitrary edge of  $E'$
5.      $C \leftarrow C \cup \{u, v\}$
6.     Remove from  $E'$  every edge incident on either  $u$  or  $v$
7. **Return**  $C$

The algorithm works as follows. We initially set the vertex cover  $C$  to null and  $E'$  to the edge set of graph  $G$ . In each iteration, we pick an arbitrary edge from  $E'$ , add its two vertices  $u, v$  to  $C$  and delete from  $E'$  all edges incident on  $u$  or  $v$ . The running time of the algorithm is  $O(V + E)$ , using adjacency lists to represent  $E'$

**Theorem 1.** APPROX-VERTEX-COVER is a polynomial-time 2-approximation algorithm.

*Proof.* We have already seen that the algorithm runs in polynomial time.

The set of vertices,  $C$  that is returned by the algorithm is a vertex cover as we iterate until every edge in  $E[G]$  is covered by some vertex in  $C$ .

Let the minimal vertex cover be  $C^*$  and let  $A$  be the set of edges that were selected in step 4 of the APPROX-VERTEX-COVER algorithm. We can observe that no two edges in  $A$  share the same endpoint, as in step 6 we remove from  $E'$  all edges that shares a vertex with the edge  $(u, v)$  selected in step 4. Therefore,  $C^*$  must include one vertex of each edge in  $A$ .

$$\Rightarrow |C^*| \geq |A| \tag{2}$$

As the vertex cover  $C$  is formed by adding vertices of all edges selected in step 4

$$|C| = 2|A| \tag{3}$$

From equations (2) and (3)

$$|C| \leq 2|C^*| \tag{4}$$

□

## 4 Travelling Salesperson Problem (TSP)

The Travelling Salesperson Problem is one of the most intensively studied problems in optimization. In its graph form, given a complete undirected graph  $G(V, E)$  that has a non-negative cost  $c(u, v)$  associated with each edge  $(u, v)$ , the problem is to find a hamiltonian cycle (or tour) of  $G$  with minimum cost. In more general terms, the problem is to find a minimum cost cycle that will traverses all vertices without traversing the same vertex twice. If  $A$  is the set of edges traversed,

$$c(A) = \sum_{(u,v) \in A} c(u,v)$$

We assume that the triangular inequality condition  $c(u,v) \leq c(u,w) + c(w,v)$  holds true as this is a natural with most practical applications. The travelling sales person problem has been proven to be NP-complete even with this assumption and in this section we examine a 2-approximation TSP Algorithm.

APPROX-TSP-TOUR( $G$ )

1. Select a vertex  $r \in V[G]$  to be the "root" vertex
2. Compute a minimum spanning tree  $T$  for  $G$  from root  $r$
3. Let  $L$  be the list of vertices visited in a preorder tree walk of  $T$
4. Return the hamiltonian cycle  $H$  that visits the vertices in the order  $L$

Using Prim's algorithm implemented with an adjacency matrix to find the minimum spanning tree will take  $O(V^2)$  time thus making APPROX-TSP-TOUR a polynomial-time algorithm.

**Theorem 2.** APPROX-TSP-TOUR is a polynomial-time 2-approximation algorithm.

*Proof.* We have already seen that the algorithm runs in polynomial time.

Let  $H^*$  be an optimal tour for the given set of vertices and let  $T$  be the spanning tree obtained in step 2 of the APPROX-TSP-TOUR algorithm. From the property of the minimum spanning tree and the observation that the optimal tour  $H^*$  is a tree (without the final loop-back edge), we can say

$$c(T) \leq c(H^*) \tag{5}$$

A full walk  $W$  of the tree  $T$  would go over each edge twice as depicted in Figure 2 (c). Therefore,

$$c(W) \leq 2c(T) \tag{6}$$

and list of vertices visited during the walk are

$$a, b, c, b, h, b, a, d, e, f, e, g, e, d, a \tag{7}$$

So cost of  $W$  is within a factor of 2 of the cost of an optimal tour but as  $W$  visits some vertices more than once, we cannot call it a tour. However, using the triangular inequality, we can remove the duplicate vertices without increasing the cost, for example we can replace  $c \rightarrow b \rightarrow h$  with  $c \rightarrow h$ . Applying this repeatedly over (7) gives us

$$a, b, c, h, b, d, e, f, g \tag{8}$$

This ordering is the same as that obtained by the preorder traversal in step 3 of APPROX-TSP-TOUR algorithm. Let  $H$  be the cycle corresponding to this pre-order traversal. It is a hamiltonian cycle as every vertex is visited exactly once. Due of the triangular inequality property discussed earlier

$$c(H) \leq c(W) \tag{9}$$

From equations (5), (6) and (9)

$$c(H) \leq 2c(H^*) \tag{10}$$

□

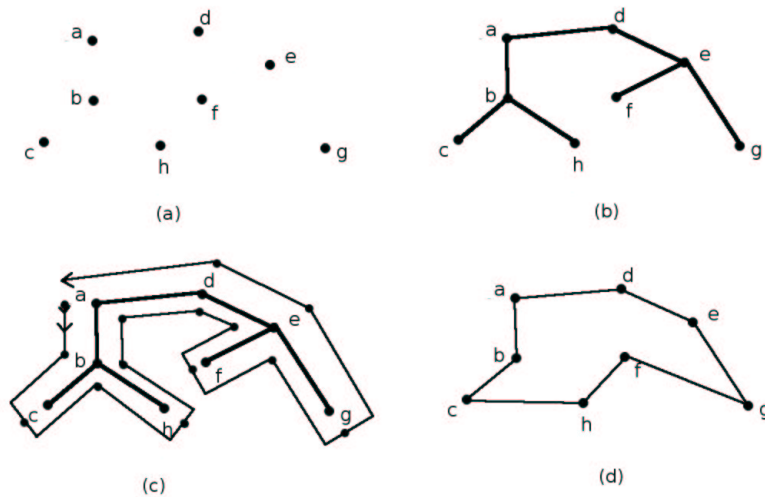


Figure 2: (a) Given set of points. (b) Minimum Spanning Tree (c) Full walk W along the tree in b (d) The tour H returned by the algorithm

## 5 Concluding remarks

In this document, we have explored the basic concepts of approximation algorithms and have examined 2-approximation algorithms for two well studied optimization problems that are NP-Complete viz. The Vertex Cover Problem and The Travelling Salesperson Problem. Though the APPROX-TSP-TOUR algorithm has a good constant-approximation, other algorithms exist which work better in practice. Also note that, without assuming the triangular inequality property, the approximation of Travelling Salesperson Problem becomes NP-Hard.

## References

- [1] T.H.Cormen, T.H.Lieserson, R.L.Rivest, C.Stein. Introduction to Algorithms, Chapter 35.
- [2] Wikipedia - The Free Encyclopedia. Traveling Salesman Problem.
- [3] D. J. Rosenkrantz, R. E. Stearns and P. M. Lewis, An Analysis of Several Heuristics for the Traveling Salesman Problem, SIAM Journal on Computing, 6, 3, Sept. 1977, 563-581.
- [4] Christos H. Paapdimitriou and Kenneth Steiglitz. Combinational Optimization: Algorithms and Complexity. Prentice-Hall, 1982.