CSCE 686 Homework 5 - SCP Design

Jon Knapp and Justin Fletcher

April 23, 2016

1 Talbi 1.5 [a]

Show that it is easy to find a simple greedy heuristic that guarantees a 2-approximation factor for the minimum vertex covering problem.

Given a graph, $G = (V, E)$, the vertex cover is a set of vertices such that every edge touches, or is incident to, at least one of them [1]. An unweighted minimum vertex cover is, is the smallest possible set of vertices which covers all edges [2], and is formally defined as

$$I \subseteq V(G)$$

where,

$$\min |I|$$
$$s.t.\ E(G) \subseteq \bigcup_{u,v \in I} \{uv\}.$$

If the vertices are weighted, and the minimum-weight vertex cover is desired, then the minimization constraint becomes

$$\min \sum_{v \in I} w(v).$$

Without a heuristic, this problem is NP-Hard. In order to approximate the solution, we introduce a matching in G. A matching is a set of edges such that no two edges are incident to a common vertex [3]. A vertex is thus matched, or saturated, if an edge in the matching is incident to that vertex.

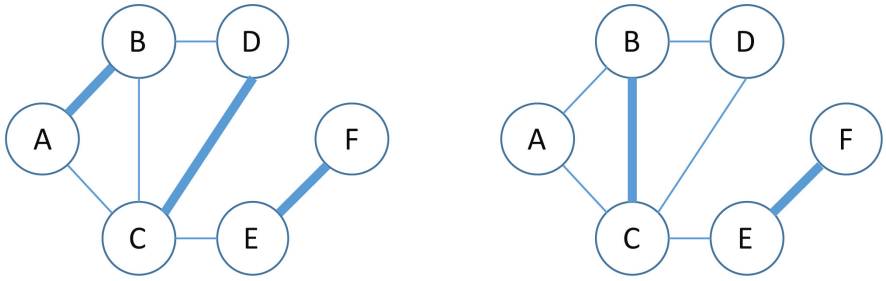

Figure 1: This figure displays two examples of maximal matching, shown as bold edges.

A maximal matching M is obtained when no further edges can be added to M without violating the matching constraint.

The maximal matching is related to the vertex cover [3], in that a valid vertex cover is also the set of all endpoints in any maximal matching M. To illustrate this concept, two graphs are shown in Figure 1, where the set of endpoints for any maximal matching, depicted as bold lines, also forms a valid vertex cover.

If we assume that some vertex set $I \subseteq V(G)$ is the minimum vertex cover for G, then we also know that $|I| \geq |M|$, where M is any maximal matching. I must contain at least one of vertices to which every edge in M is incident, in order to cover all edges in G. Since no more edges can be added to a maximal matching, all remaining edges are adjacent to at least one edge in M. We can cover the entire graph by choosing the endpoint vertices to all edges in M.

We define an algorithm, $GetM$, to produce M, which is included as Algorithm 1 in this work. The algorithm produces a maximal matching. All pairs of vertices that are added into C are unique, and thus eliminating any edges that share a vertex. The algorithm only terminates when there are no uncovered edges. Using this relationship, we conclude that the following:

$$|I| \geq |M| = GetM(G)/2$$

The vertex cover is size $2M$. The previous relationship can be written as:

Algorithm 1 This algorithm describes 2-approximation heuristic. *Inputs:* E, a set of edges. *Outputs:* Set of vertices C.

```
1: procedure GET2APPROX(E)
2:     C ← φ
3:     while (u, v) not covered, and u ≠ v do        ▷ Iterate over every potential edge
4:         Add u to C.
5:         Add v to C.
6:     end while
7: end procedure
```

$$GetM(G) \leq 2|I|$$

The heuristic that we presented uses the relationship between matching sets and the set covering problem. By finding a maximal matching, we can 2-approximate the minimum vertex cover problem with $O(m)$ complexity.

2 Talbi 1.11 [a]

Define one or more greedy algorithms for the CVRP problem. Give some examples of constraints or more general models encountered in practice.

The Capacitated Vehicle Routing Problem (CVRP) is defined by a Graph $G = (V, A)$. The nodes represent "customers" and A defines costs, denoted as c_{ij}, associated with each edge. Additionally, a central "depot" node is defined, as well as a capacity Q for each "vehicle," associated with each "route" [ref MCT1]. A depot can only have H number of routes. Customers have an associated demand, d_i, which defines the quantity that will be used to calculate the total capacity of a route. This problem finds routes for a fleet of vehicles so that every customer is covered and no vehicle exceeds its capacity.

A well-known algorithm for solving the CVRP is the Clarke and Wright Savings (CWS) algorithm, developed in 1964 [2]. It is a greedy, heuristic driven algorithm that does not guarantee an optimal solution, although it

will often provide a good solution [3]. This method is initialized by assuming that each customer (non-depot node), is visited by its own vehicle. If the graph consists of the depot node A, and two customers i and j, the cost $D1$, is calculated using the formula

$$D1 = c_{Ai} + c_{iA} + c_{Aj} + c_{jA}$$

The algorithm combines routes to obtain a new route by assigning two customers to be served by the same vehicle, assuming the vehicle's capacity Q has not been exceeded. The new cost is

$$D2 = cAi + cij + cjA$$

A total cost savings, S_{ij}, is calculated:

$$Sij = D1D2 = ciA + cAjcij$$

This saving is used to determine if the points i and j should be on the same route. Large values of S indicate that the route is advantageous. The algorithm starts by calculating the savings for all pairs of customers and putting them in a sequence R. This sequence is then sorted in descending order of the savings. When considering a pair of points i and j whose capacity is less then Q for a particular route, they are added to the route such that i is visited and then j, but only if this can be done without disrupting the direct connection between two nodes. In the parallel version of this algorithm, only one pass is required through R, and routes are built simultaneously [3].

Not only does this algorithm not guarantee an optimal solution, it cannot guarantee that the number of routes will be exactly H. A particular execution of the algorithm may find some $H\prime$ greater then H. However, using the savings heuristic it can produce a solution that is usually close to optimal [4].

There are a wide variety of solutions to the CVRP in the literature that use Monte-Carlo techniques to improve the search. One such algorithm, presented by [ref 2], uses a random sampling to acquire a solution.

In practice, there are many additional constraints for CVRP problems. One such domain is the oil and gas industry [4]. For particular delivery vehicles, the presence of flow meters can be used to satisfy the demands of multiple customers. Additionally, after certain operations, the delivery vehicles may require cleaning, which must be factored into the route plan. Although not specifically mentioned in the source, the presence of refueling stations would constrain the routes further.

Another domain that can constrain VRP problems is retail. For any particular product supplier, the cost of delivering goods to stores is a significant investment. This cost can be reduced with CVRP algorithms. From personal experience in the industry, we use the floral industry as a hypothetical example. The constraints for this application include not combining certain products (some flowers and plants cannot be shipped together), always sending certain drivers to a particular customer (customer loyalty), delivering highly perishable goods first, or long lived items last. Additional considerations include how profitable a particular customer is and the risk associated with delivering late products, which could affect the likelihood of including a customer in a highly constrained route.

3 Design for the Set Cover Problem [c.1]

In this section, we discuss the design of the Set Covering problem (SCP) as introduced in [**?**].

3.1 Problem Domain Description: SCP

Given a set $R = \{r_1, ..., r_m\}$, and a family $\mathcal{F} = \{S_1, ..., S_N\}$ of sets $S_j \subset R$ any sub-family $\mathcal{F}' = \{S_j1, S_j2, ..., S_jn\}$ of $\mathcal{F}$, a set covering of R is defined as

4 Testing the AFIT SCP Solver [c.2]

In this section, we describe the performance of the AFIT SCP Program (ASP) when applied to several SCP problems of varying complexity. The performance of the ASP is evaluated on a real-world Air Force problem, and

is rigorously examined using a set of contrived SCP instances; these instances are constructed such that many combinations of number of cover-elements, number of covering sets, and density are considered.

4.1 The RIF Problem: An Air Force Application of SCP [c.3.1]

In order to evaluate the performance of the ASP on a real-world[1] problem, we define a problem which is relevant to the United States Air Force (USAF). The USAF has a budget problem; in order to solve this problem it has decided to fire a large portion of its workforce. Though this problem is applicable to the USAF as a whole, we limit the examination of the problem in this work to the community of pilots, for ease of analysis. The elements which must be covered in this problem are the individual aircraft to be flown. The covering sets are the pilots, each of which can fly at least one jet; for simplicity we assume that the cost of a pilot is directly proportional to their pay grade.

4.2 Rigorous Testing [c.3.2]

In this section we evaluate the performance of the ASP on problems of varying dimensionality. An SCP instance can be defined abstractly with three quantities, the number of sets, the number of cover elements, and the density. Because the complexity of the problem depends on all three of these quantities, we vary all three in our experiments in this work. ... In these figures three plots are displayed, each of which highlights a particular element of the algorithms performance. The top-right plots are tile plots which show the mean running time, over 20 runs, of the SCP program, for the number number of sets indicated on the horizontal axis, and the number os elements indicated on the vertical axis. The bottom-right plot displays the marginal mean and standard deviation of program running time, as the number of sets varies. These values are calculated by taking the mean and standard deviation of the mean running time values for a particular number of sets, across all numbers of elements to cover. The top-left plot shows the marginal mean and standard deviation of running time

[1] This problem is only notionally real-world, and is constructed for the purposes of this work.

References

[1] Lecture21, carnegie mellon university, school of computer science. [Online]. Available: http://www.cs.cmu.edu/ avrim/451f12/lectures/lect1106.pdf

[2] Cs105 notes, dartmouth. [Online]. Available: http://www.suhendry.net/blog/?p=1647

[3] 2-approximation for minimum vertex cover problem. [Online]. Available: http://www.suhendry.net/blog/?p=1647