A NEW ALGORITHM FOR THE RESOURCE CONSTRAINED SHORTEST PATH PROBLEM

Cenk Çalışkan, Woodbury School of Business, Utah Valley University, 800 W. University Pkwy, Orem, UT 84058, (801) 863-6487, cenk.caliskan@uvu.edu

ABSTRACT

The resource constrained shortest path problem is to find a minimum cost path from an origin node to a destination node on a directed network subject to a set of side constraints. The resource constrained shortest path problem is NP-complete, so there is no algorithm that can solve it in polynomial time. The problem has applications in many areas of business such as in transportation, logistics, supply chain management; as well as in computer networks, telecommunications and other areas of science and engineering. It usually appears as a subproblem as well, so it is important to solve it efficiently. We propose a promising new algorithm that is based on Lagrangian relaxation, subgradient optimization and branch-and-bound methods. Our method solves a series of unconstrained shortest path problems, which can be solved efficiently. This, together with tight Lagrangian lower bounds makes our method particularly attractive.

Keywords: shortest path problem, lagrangian relaxation, networks, side constraints, subgradient optimization

INTRODUCTION

The resource constrained shortest path problem is an important problem in operations research. It has applications in many areas of business such as in transportation, logistics, supply chain management; as well as in computer networks, telecommunications and many other areas of science and engineering. For example, we may wish to find a minimum cost route subject to a total time constraint in a multi-mode transportation network. The problem also frequently appears as a subproblem in more complicated problems, therefore it is important to solve it efficiently. The problem is shown to be at least as hard as NP-complete problems and it is generic to a class of problems that arise in the solution of integer linear programs and discrete state/stage deterministic dynamic programs.

The resource constrained shortest path problem is a variant of the classical shortest path problem in which the objective is to find a lowest cost path from an origin node to a destination node on a directed network ([10], [7], [13]). This problem is easily solvable in polynomial time and it is one of the relatively small subset of network flow problems that have this property ([17], [12], [8], [1]). Computational comparisons of the existing shortest path algorithms using randomly generated networks are provided in [4] and using actual road networks, in [20].

Many constrained variations of the classical shortest path problem exist. As an example, the path may be constrained to include specific nodes, or be constrained to include a specific number of nodes (e.g., see [9]), or include nodes within a pre-specified covering distance of every node in the

network ([6, 5]). In general, the term ``constrained shortest path problem" refers to the problem in [15] that includes one side constraint that establishes an upper limit on the sum of some other arc cost for the path. Unfortunately, the addition of such constraints to the shortest path problem generally results in a problem that belongs to the set of problems known as NP-hard (e.g., see [14]).

One approach to solve it is utilizing a k-shortest path algorithm and terminating the algorithm when a feasible path is obtained. However, this approach is not computationally attractive when the terminal value of k is large. Several algorithms have been developed to solve the k-shortest path problem (e.g., [11], [18] and [2]). [3] is the improved version of the k-shortest path algorithm proposed in [2]. [15] uses a Lagrangian relaxation approach that reduces the value of the k needed to solve the constrained shortest path problem. Other approaches based on dynamic programming (e.g., [16]) have been proposed. However, the Handler and Zang Lagrangian relaxation-based method is generally considered the most efficient to date. [19] modifies the Lagrangian relaxation approach proposed in [15] and reports improved computational performance.

In this paper, we propose another computationally attractive Lagrangian relaxation based procedure, but our approach uses a branch and bound technique. The solution procedure involves solving a series of progressively smaller unconstrained shortest path problems, which can be efficiently accomplished in practice.

PROBLEM DESCRIPTION

Let $G=(N, A)$ be a directed network consisting of a set N of nodes and a set A of arcs, where $|N| = n$ and $|A| = m$. In this network, an arc from node i to node j is denoted by (i, j). The cost on arc (i, j) is denoted by c_{ij}. Node s is called the source, and node t is called the sink node in this network. There are p resources and the amount available for resource k is denoted by s_k. An arc (i, j) requires q_{ijk} units of resource k if it is included in the chosen path. The resource constrained shortest path problem is then to find a minimum cost path subject to constraints on resource usage. The problem can be expressed as the following binary integer programming problem:

$$z = \min \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{1}$$

s.t.

$$\sum_{(i,j)\in A} x_{ij} - \sum_{(j,i)\in A} x_{ji} = \begin{cases} 1 & \text{for} \quad i = s \\ 0 & \text{for all} \quad i \in N \setminus \{s,t\} \\ -1 & \text{for} \quad i = t \end{cases} \tag{2}$$

$$\sum_{(i,j)\in A} q_{ijk} x_{ij} \le s_k \quad \forall k \in \{1,2,\ldots,p\} \tag{3}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in A \tag{4}$$

Eq. 1 is the total cost of a feasible path from s to t. Eqs. 2 are the conservation of flow constraints that ensure that a feasible solution x defines a path from s to t. Eqs. 3 are the resource constraints and Eqs. 4 are the binary restrictions on the variables.

LAGRANGIAN RELAXATION METHOD

If we relax the resource constraints (Eq. 3), the resulting problem is a classical shortest path problem. As it is possible to efficiently solve shortest path problems, this makes the Lagrangian relaxation approach particularly attractive. We define the Lagrange multipliers λ_k, for $k = 1,2,3,\ldots, P$ corresponding to the resource constraints (Eq. 3). Let X be the set of x that satisfy Eqs. 3 and 4. We then relax the resource constraints and bring them into the objective function, resulting in the following Lagrangian subproblem or Lagrangian function:

$$L(\lambda) = \min_{\lambda \geq 0} \{ \sum_{(i,j) \in A} c_{ij} x_{ij} + \sum_{k=1}^{p} \lambda_k (\sum_{(i,j) \in A} q_{ijk} x_{ij} - s_k) : x \in X \}.$$

This can be simplified into the following equation and it is obviously a shortest path problem for a given vector of values for the Lagrange multipliers:

$$L(\lambda) = \min_{\lambda \geq 0} \{ \sum_{(i,j) \in A} (c_{ij} + \sum_{k=1}^{p} \lambda_k q_{ijk}) x_{ij} - \sum_{k=1}^{p} s_k \lambda_k : x \in X \} \quad (5)$$

Let $\delta_k = \sum_{(i,j) \in A} q_{ijk} x_{ij} - s_k$ for $k = 1,2,\ldots, p$. Then, Eq. 3 could be re-written in the form $\delta \leq 0$.

Lemma 1 *For any $\lambda \geq 0$, the value $L(\lambda)$ of the Lagrangian function is a lower bound on the optimal objective function value z^* of the original constrained shortest path problem.*

Proof: Let x^* be an optimal solution to the constrained shortest path problem and $\hat{x}$ ia an optimal solution to the Lagrangian subproblem. Then,

$$z^* = cx^* \geq cx^* + \lambda(Qx^* - s) \geq c\hat{x} + \lambda(Q\hat{x} - s) \quad (6)$$

The first part of this inequality is because x^* is a feasible solution to the constrained shortest path problem, and the second part is because $\hat{x}$ is an optimal solution to the Lagrangian subproblem.□

In order to obtain the tightest lower bounds, we maximize the Lagrangian function, i.e. we solve the following maximization problem, which is called the Lagrangian dual:

$$L^* = \max_{\lambda \geq 0} L(\lambda). \quad (7)$$

In order to solve the Lagrangian dual, we use a subgradient optimization procedure. We use an adaptation of the Newton's method in which we update the Lagrange multipliers in a relatively large step towards the optimal solution along the subgradient direction. The initial value of the multipliers are set to zero, and at every iteration, they are updated as follows:

$$\lambda_k^{t+1} = \max\{0, \lambda_k^t + \theta_t \delta_k\} \quad \forall k \in \{1,2,\ldots, p\} \quad (8)$$

where θ_t is the step size at iteration t. In order to make sure that the procedure will converge, the step size is chosen as follows (see, for instance, [1]):

$$\theta_t = \frac{\sigma_t [UB - L(\lambda^t)]}{\| \delta \|^2} \quad (9)$$

where $\| \delta \| = (\sum_k \delta_k^2)^{0.5}$ is the Euclidean norm of δ and σ_k is a scalar that is chosen strictly between 0 and 2. the initial value of σ is 2, and it is halved once the Lagrangian objective function fails to increase after a long series of iterations. UB is the surrogate for the optimal

Lagrangian objective function value, L^* and it is chosen as the best upper bound, i.e. the total cost of the lowest cost feasible $s-t$ path found thus far in the algorithm.

procedure *solve_lagrangian_dual*(Γ);
begin
  $z := \sum_{(i,j)\in P} c_{ij}$;
  $\lambda := 0$;
  $\sigma := 2$;
  $count := 0$;
  while P is not a feasible path **and** $z < UB$ **and** $count < T_{max}$ **do**
    $\theta := \sigma[UB - L(\lambda)] / \| \delta \|^2$;
    $\lambda := \lambda + \theta\delta$;
    $c_{ij} := c_{ij} + \sum_{k=1}^{p} \lambda_k q_{ijk} \quad \forall (i,j) \in A$;
    P := Shortest path from s to t in Γ;
    $z := \sum_{(i,j)\in P} c_{ij}$;
    $count := count + 1$;
    if z has not changed in T_{lim} iterations **then**
      $\sigma := \sigma/2$;
    end if
  end while
  return (z, P);
end

Figure 1: The procedure *solve_lagrangian_dual* of the branch and bound algorithm

The pseudocode for the subgradient method to solve the Lagrangian dual problem is given in Figure 1. Note that the iterations of the subgradient method may be stopped whenever $L(\lambda) \geq UB$ as that means we cannot obtain a better feasible tour than the best existing tour from this branch and bound node. We describe the branch and bound algorithm in the next section. The pseudocode for the branch and bound algorithm is shown in Figures 2-3.

THE BRANCH AND BOUND ALGORITHM

We start the branch and bound procedure by solving the Lagrangian dual for the original network using the subgradient method. If the resulting solution is a feasible path, we might have obtained the optimal solution. In order to determine if we have obtained the optimal path, we rely on the following lemma:

Lemma 2 (Optimality Conditions) *A solution* (x^*, λ^*) *to the Lagrangian subproblem* $L(\lambda)$ *is also an optimal solution to the constrained shortest path problem if and only if:*

(i) $Qx^* \leq s$

(ii) $\lambda^*(Qx^* - s) = 0$ (Complementary slackness)

Proof: Condition (i) implies that x^* is a feasible path. Since $L(\lambda)$ is a lower bound for the optimal value of the constrained shortest path problem, we have: $cx^* \geq L(\lambda^*) = cx^* + \lambda(Qx^* - s)$. By condition (ii), we have $cx^* = L(\lambda^*)$. □

If this initial solution to the Lagrangian subproblem does not satisfy the optimality conditions, then it is either infeasible or feasible but does not satisfy complementary slackness. We then branch on one of the arcs. If the solution is feasible, then we branch on the arc $(u, v) \in P$ with the highest c_{uv}. If the solution is infeasible, then we branch on the arc that is determined as follows:

$$(u,v) := argmax_{(i,j)}(\sum_{k=1}^{p} q_{ijk} \cdot \max\{0, \delta_k\}) \tag{10}$$

Let us assume that we are branching on arc (i, j). We will create two branches: one in which the arc cannot be used for going from i to j and the other in which the arc has to be used for going from i to j. The former can be accomplished by removing it from the network, whereas the latter can be accomplished by collapsing nodes i and j into one node. Clearly, the two branches are mutually exclusive. We then apply the subgradient method to one of the branches that we pick and keep repeating this until all of the created branches are fathomed at which time we will have found the optimal solution to the resource constrained shortest path problem. If any of the branches has a lower bound greater than the cost of the best feasible path, that branch is fathomed because it cannot yield a better solution. If any of the branches yields a feasible path and satisfies Lemma 2, then that branch is fathomed as well, as we can't find a better solution from that branch. We update the best UB if this feasible solution has a lower total cost.

procedure *make_branches(P)*;
begin
  if P is feasible **then**
    $(u,v) := argmax_{(i,j)}(c_{ij}, \forall (i,j) \in P)$
  else
    $(u,v) := argmax_{(i,j)}(\sum_{k=1}^{p} q_{ijk} \cdot \max\{0, \delta_k\})$
  end if
  G_1 := G;
  G_2 := G;
  { *Create branch 1*}
  Remove (u,v) from G_1;
  { *Create branch 2*}
  Collapse u and v into one node in G_2 and add c_{uv} to the fixed cost of G_2;
  $\mathsf{B} := \mathsf{B} \cup \{G_1\}$;
  $\mathsf{B} := \mathsf{B} \cup \{G_2\}$;
end

Figure 2: The procedure *make_branches* of the branch and bound algorithm

CONCLUSION

In this research we propose a Lagrangian relaxation based branch and bound procedure to solve the resource constrained shortest path problem. At each node of the branch and bound tree, subgradient optimization method is applied to solve the Lagrangian dual problem to obtain a lower bound. The proposed method is especially attractive as the entire solution process reduces to a

algorithm *branch_and_bound*;
begin
$UB := \infty$;
P := Shortest path from s to t in G
if P is a feasible path **then**
Stop. P is optimal and $z^* := \sum_{(i,j)\in P} c_{ij}$;
else
$\mathsf{B} = \varnothing$
make_branches;
while $\mathsf{B} \neq \varnothing$ $\hat{G} :=$ a branch and bound node from B **do**
$\{LB, P\} :=$ *solve_lagrangian_dual*$(\hat{G})$
if P is a feasible path **then**
$UB := \min\{UB, \sum_{(i,j)\in P} c_{ij}\}$;
if x, λ do not satisfy Lemma 2 **then**
make_branches;
end if
$\mathsf{B} = \mathsf{B} \setminus \hat{G}$
if $LB < UB$ **then**
make_branches;
end if
$\mathsf{B} = \mathsf{B} \setminus \hat{G}$
end if
end while
end if
end

Figure 3: The algorithm *branch_and_bound*

series of unconstrained shortest path problems on progressively smaller networks, which can be solved efficiently. It is also attractive due to the fact that Lagrangian relaxation lower bounds are typically tighter than LP relaxation lower bounds. We describe the details of the algorithm in this paper. Future research will focus on computational testing of the proposed solution procedure as compared to the existing methods for the constrained shortest path problem.

REFERENCES

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network flows: Theory, algorithms and applications*. Prentice-Hall, Englewood Cliffs, NJ, 1993.
[2] J.A. Azevedo, M. Costa, J. Madeira, and E.Q.V. Martins. An algorithm for the ranking of shortest paths. *European Journal of Operational Research*, 69:97–106, 1993.
[3] J.A. Azevedo, J. Madeira, M. Costa, E.Q.V. Martins, and F. Pires. A computational improvement for a shortest paths ranking algorithm. *European Journal of Operational Research*, 73:188–191, 1994.
[4] B.V. Cherkassky, A.V. Goldberg, and T. Radzik. Shortest paths algorithms: Theory and experimental evaluation. *Mathematical Programming*, 73(2):129–174, 1996.
[5] J.R. Current, H. Pirkul, and E. Rolland. Efficient algorithms for solving the shortest covering path problem. *Transportation Science*, 28(4):317–327, 1994.
[6] J.R. Current, C.S. ReVelle, and J.L. Cohon. The shortest covering path problem: an application of locational constraints to network design. *Journal of Regiaonal Science*, 24:161–185, 1984.
[7] George B. Dantzig. On the shortest route through a network. *Management Science*, 6(2):187–190, 1960.
[8] M. Daskin. *Network and Discrete Location*. John Wiley and Sons, New York, 1995.
[9] N. Deo and C.Y. Pang. Shortest-path algorithms: Taxonomy and annotation. *Networks*, 14(2):275–323, 1984.
[10] E.W. Dikjstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1:269–271, 1959.
[11] S. Dreyfus. An appraisal of some shortest path algorithms. *Operations Research*, 17:345–412, 1969.
[12] J.R. Evans and E. Minieka. *Optimization Algorithms for Networks and Graphs*. Marcel Dekker, New York, 1992.
[13] R.W. Floyd. Algorithm 97: Shortest path. *Commun. ACM*, 5(6):345, June 1962.
[14] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, New York, 1979.
[15] G.Y. Handler and I. Zang. A dual algorithm for the constrained shortest path problem. *Networks*, 10(4):293–309, 1980.
[16] H.C. Johsch. The shortest route problem with constraints. *Journal of Mathematical Analysis*, 14:191–197, 1966.
[17] T.L. Magnanti and R.T. Wong. Network design and transportation planning: Models and algorithms. *Transportation Science*, 18(1):1–55, 1984.
[18] E.Q.V. Martins. An algorithm for ranking paths that may contain cycles. *European Journal of Operational Research*, 18:123–130, 1984.
[19] L. Santos, J. Coutinho-Rodrigues, and J.R. Current. An improved solution algorithm for the constrained shortest path problem. *Transportation Research Part B: Methodological*, 41(7):756 – 771, 2007.
[20] F.B. Zhan and C.E. Noon. Shortest path algorithms: An evaluation using real road networks. *Transportation Science*, 32(1):65–73, 1998.