MIT/LCS/TM-457

AN ALGORITHM FOR THE TRAMP STEAMER PROBLEM BASED ON MEAN-WEIGHT CYCLES

Alexander T. Ishii
Charles E. Leiserson
Marios C. Papaefthymiou

November 1991

An Algorithm for the Tramp Steamer Problem Based on Mean-Weight Cycles

Alexander T. Ishii
Charles E. Leiserson
Marios C. Papaefthymiou

Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

November 4, 1991
(Revised December 1, 1991)

Abstract

Given a connected, directed graph $G = (V, E)$ in which each edge (u, v) has a cost $c(u, v)$ and a transit time $t(u, v)$, the tramp steamer problem is defined as identifying a directed cycle within the graph for which the ratio of total cost to total transit time is as small as possible. We give an $O(VE + t^*E)$-time[1] algorithm for the problem, where t^* is the transit time around the optimal cycle. Our algorithm assumes, as is common in the literature, that transit times are integer valued and that the total transit time around any cycle is positive. Our algorithm does not scale costs and thus avoids the common assumption that costs are integer valued. If the scaling of costs is acceptable, however, our algorithm can be adapted to run in $O(V^{1/2}E \lg(Vc_{\max}t_{\max}) + t^*E)$-time, where $t_{\max} = \max_{(u,v)\in E} |t(u, v)|$ and $c_{\max} = \max_{(u,v)\in E} |c(u, v)|$. Among algorithms that do not scale costs, our procedure is the asymptotically fastest to date for the problem when $t^* = o(V \lg(Vc_{\max}t_{\max}))$. When costs are scaled, the adapted version of the algorithm is the asymptotically fastest to date for the problem when $t^* = o(V^{1/2} \lg^2(Vc_{\max}t_{\max}))$.

Keywords: tramp steamer problem, minimum cost-to-time ratio, minimum cycle mean, combinatorial optimization, algorithms.

1 The Problem

The *tramp steamer problem* (also known as the *minimum cost-to-time ratio cycle problem*) was formulated by Dantzig, Blattner, and Rao [2] as follows. Let $G = (V, E)$ be a directed graph in which each edge (u, v) has an integer cost $c(u, v)$ and an integer transit time $t(u, v)$, such that for any cycle C in G, we have $\sum_{(u,v)\in C} t(u, v) > 0$. For any cycle C in G, define the cost-to-time ratio of the cycle by

$$R(C) = \frac{\sum_{(u,v)\in C} c(u, v)}{\sum_{(u,v)\in C} t(u, v)} .$$

The problem is to identify a cycle C^* in G such that the cost-to-time ratio $R(C^*)$ is minimum.

The motivation for the tramp-steamer problem was provided by Dantzig *et al.* Each vertex is a port of call for a ship, and a voyage from port u to port v earns $p(u, v)$ dollars of profit and takes $t(u, v)$ days to complete. The captain of the ship wishes to determine a circular route among the ports that over

[1]For the sake of readability, inside asymptotic notation, such as O-notation, or inside algebraic expressions, we adopt the convention that the symbol V denotes $|V|$ and the symbol E denotes $|E|$.

time will guarantee him the maximum profit per day. In the graph formulation above, we simply let $c(u,v) = -p(u,v)$, and the captain wishes to find a cycle C^* in the graph for which $R(C^*)$ is minimized.

Several algorithmic techniques have been developed for the solution of the tramp steamer problem [2, 8, 9]. The asymptotically best algorithms to date perform a binary search over the space of possible optimal ratios. The search requires the solution of $O(\lg(Vc_{\max}t_{\max}))$ shortest-paths problems, where $c_{\max} = \max_{(u,v)\in E} |c(u,v)|$ and $t_{\max} = \max_{(u,v)\in E} |t(u,v)|$. Using the Bellman-Ford algorithm for shortest-paths, a solution can be obtained in $O(VE\lg(Vc_{\max}t_{\max}))$ steps [9]. A solution can be obtained in $O(V^{1/2}E\lg^2(Vc_{\max}t_{\max}))$ steps using the Gabow-Tarjan scaling algorithm for shortest-paths [3]. Another algorithm with running time $O(V^{1/2}E\lg^2(Vc_{\max}t_{\max}))$ can be obtained by using the Orlin-Ahuja scaling algorithm for shortest-paths [10]. Also, for the restricted case when transit times are either 0 or 1, the problem can be solved in $O(VE)$ time [4].

In this paper we describe an algorithm for the tramp steamer problem that performs a search for the transit time t^* around the optimal cycle. The search requires the solution of $O(\lg(t^*/V))$ shortest-paths problems. Using the Bellman-Ford algorithm for shortest-paths, the search can be performed in $O(VE+t^*E)$-time. Unlike other algorithms in the literature, the running time of this algorithm does not depend on the sizes of $c(u,v)$, and hence the typical assumption that costs are integer is unnecessary. Our procedure is based on Karp's $O(VE)$-time algorithm for finding the minimum cycle mean of a directed graph [6] and is asymptotically the fastest algorithm to date for the tramp steamer problem when $t^* = o(V\lg(Vc_{\max}t_{\max}))$ and costs are not scaled. This bound on t^* arises in various applications, including a problem involving the optimization of level-clocked circuitry [5].

If scaling of costs is acceptable, our algorithm can be adapted to run in $O(V^{1/2}E\lg(Vc_{\max}t_{\max}) + t^*E)$ steps. The adapted algorithm requires the solution of $O(\lg(t^*/V^{1/2}\lg(Vc_{\max}t_{\max})))$ shortest-paths problems. Using either of the scaling algorithms for shortest-paths mentioned above, the claimed running time is obtained, thus resulting in the asymptotically fastest algorithm to date for the problem when $t^* = o(V^{1/2}\lg^2(Vc_{\max}t_{\max}))$.

The choice of transit time, as opposed to cost, as the "search variable" is not mandatory. If the edge costs are integer and their sum around any cycle in G is positive, then we can obtain symmetric results by regarding $-t(u,v)$ as the "cost" and $c(u,v)$ as the "transit time."

2 The Algorithm

In this section we describe our algorithm for the tramp steamer problem. We assume that all edge transit times are nonnegative. If this is not true, we apply the following well-known reweighting transformation [1, Sec. 26.3]. We assign to each edge $(u,v) \in E$ a new transit time $t'(u,v) = t(u,v) + l(u) - l(v)$, where for all $u \in V$, the value $l(u)$ denotes the minimum transit time to vertex u from any vertex in V. The value $l(u)$ is well defined for all u, since the transit time around any cycle is nonnegative. It follows from the definition of l that $t'(u,v) \geq 0$ for all $(u,v) \in E$. Moreover, since the transit time around any cycle is the same with respect to either t or t', the optimal cost-to-time ratio in G remains unaltered by the transformation. The variables $l(u)$ can be computed in either $O(VE)$-time or $O(V^{1/2}E\lg(Vt_{\max}))$-time, using either Bellman-Ford or a shortest-paths scaling algorithm, as appropriate.

The basic algorithm, shown in Figure 1, makes the simplifying assumption that all edges have strictly positive transit times. The algorithm can be understood as a search over the space of possible values for t^*. Assume that a source vertex s has been introduced into G. As will be discussed in the next section, if t^+ is an upper bound on the value of t^*, the value of $R(C^*)$ can be obtained given, for each vertex v and integer $k \leq t^+$, the cost $F_k(v)$ of the least-cost path from s to v with total transit time exactly k. Since only unacceptably large upper bounds are known *a priori*, however, the algorithm begins with a suitable base estimate $n = |V|$ of an upper bound, and proceeds to double the estimate n until a true upper bound is isolated. The algorithm determines whether a particular n is a true upper bound by computing an approximation R_n of the desired cost-to-time ratio, and comparing the

```
TRAMP$(V, E, c, t)$
1  $V' \leftarrow V \cup \{s\}$, $E' \leftarrow E \cup \{(s,v) : v \in V\}$
2  for each vertex $v \in V$
3      do $t(s,v) \leftarrow 1$
4         $c(s,v) \leftarrow 0$
5  $F_0(s) \leftarrow 0$
6  for each vertex $v \in V$
7      do $F_0(v) \leftarrow \infty$
8  $n \leftarrow |V|$
9  $m \leftarrow 1$
10 while TRUE
11    do for $k \leftarrow 1$ to $n$
12           do for each vertex $v \in V'$
13                  do $F_k(v) \leftarrow \infty$
14                     for each edge $(u,v) \in E'$ with $k \geq t(u,v)$
15                         do $F_k(v) \leftarrow \min\{F_k(v), F_{k-t(u,v)}(u) + c(u,v)\}$
16                     if $F_k(v) < \infty$
17                       then $m \leftarrow k$
18       $R_n \leftarrow \infty$
19       for each vertex $v \in V'$
20           do $R_n(v) \leftarrow -\infty$
21              for $k \leftarrow 1$ to $m-1$
22                  do $R_n(v) \leftarrow \max\{R_n(v), (F_m(v) - F_k(v))/(m-k)\}$
23              $R_n \leftarrow \min\{R_n, R_n(v)\}$
24       if the graph with edge weights $c(u,v) - R_n \cdot t(u,v)$ has a negative-weight cycle
25         then $n \leftarrow 2n$
26         else return $R_n$
```

Figure 1.

approximation to $R(C^*)$. The approximation R_n is the best estimate of $R(C^*)$ that the algorithm is able to isolate when only paths of total transit time at most n are considered. Since it is not necessarily true that for every particular n there exists a path from s with transit time n, the algorithm examines paths of transit time up to the maximum $m \leq n$ that can be achieved by any path. The existence of a negative-weight cycle in the graph (V, E, w), where $w(u,v) = c(u,v) - R_n \cdot t(u,v)$, implies that $R_n > R(C^*)$ [9]. If no negative-weight cycle exists, then $R_n = R(C^*)$ and every optimal cycle has zero weight. From this point, an optimal cycle C^* can be identified by an $O(E)$-time depth-first search. The variables $R_n(v)$ are temporary variables used to compute R_n. Our choice of m guarantees that, for $n > t^*$ and some vertex v on the optimal cycle C^*, the cost $F_m(v)$ is finite and the variable $R_n(v)$ attains the value $R(C^*)$.

The basic algorithm ignores the possibility of edges with zero transit time. Such edges can be readily handled by substituting the lines of code shown in Figure 2 for lines 11–17 of Algorithm TRAMP. In order to properly compute $F_k(v)$ in the presence of zero-transit-time edges, the contribution of the costs $F_k(u)$ for all u with zero-transit-time paths to v must be considered. The code in Figure 2 accomplishes this task by propagating the effect of these values along the *acyclic* subgraph (V, E_0) of G that contains only the edges of G with zero transit time.

3 Analysis

Our algorithm is based on Karp's algorithm [6] for the minimum cycle mean problem which is defined as follows. Given a directed graph $G = (V, E)$ with weight $c(u,v)$ on each arc (u,v), identify a cycle C^*

```
11a     do $E_0 \leftarrow \{(u,v) \in E : t(u,v) = 0\}$
11b        for $k \leftarrow 1$ to $n$
12            do for each vertex $v \in V'$
13                  do $F_k(v) \leftarrow \infty$
14                     for each edge $(u,v) \in E' - E_0$ with $k \geq t(u,v)$
15a                        do $F_k(v) \leftarrow \min\{F_k(v), F_{k-t(u,v)}(u) + c(u,v)\}$
15b              for each vertex $v \in V$ in topological sort order in $(V, E_0)$
15c                 do for each edge $(u,v) \in E_0$
15d                        do $F_k(v) \leftarrow \min\{F_k(v), F_k(u) + c(u,v)\}$
16                     if $F_k(v) < \infty$
17                        then $m \leftarrow k$
```

Figure 2.

in G such that

$$M(C^*) = \frac{\sum_{(u,v)\in C^*} c(u,v)}{|C^*|} ,$$

is minimum, where $|C^*|$ denotes the number of edges in C^*. In [6], it is shown that

$$M(C^*) = \min_{v\in V} \max_{0\leq k\leq n-1} \frac{F_n(v) - F_k(v)}{n-k} ,$$

where $F_k(v)$ denotes the cost, with respect to c, of the least-cost path of k edges from some source vertex s to v.[2] The parameter n equals the number of edges in C^* plus the number k of edges required to reach the vertex v from the source s along the path corresponding to the optimal $F_k(v)$.

The correctness of our algorithm relies on the observation that for any graph where all edges have unit transit time, the minimum cost-to-time ratio is equal to the minimum cycle mean. For any given graph $G = (V, E)$, let G_{eqv} be the graph with edges of unit transit time constructed as follows. For each edge $(u,v) \in E$ with transit time $t(u,v) > 0$, the graph G_{eqv} has a sequence of edges $(u_0, u_1), (u_1, u_2), \ldots, (u_{t(u,v)-2}, u_{t(u,v)-1})$ with zero cost. For every vertex w that can be reached from v along zero-transit-time edges, G_{eqv} has an edge $(u_{t(u,v)-1}, w)$ with cost equal to $c(u,v) + l_0(v,w)$, where $l_0(v,w)$ denotes the length, with respect to c, of the shortest path in (V, E_0) from v to w. It is straightforward to verify that the minimum cost-to-time ratio of G equals the minimum cycle mean of G_{eqv}.

Algorithm TRAMP solves the minimum cost-to-time ratio problem on G by simulating the operation of Karp's minimum cycle mean algorithm on G_{eqv}. Each time a single edge (u,v) in G is processed, Algorithm TRAMP conceptually achieves in a single step the effect of running Karp's algorithm on the entire corresponding sequence of edges $(u_0, u_1), (u_1, u_2), \ldots, (u_{t(u,v)-2}, u_{t(u,v)-1})$ in G_{eqv}. The desired running-time bounds cannot be achieved by simply applying Karp's algorithm to G_{eqv}, since G_{eqv} can be larger than G by as much as a factor of $t_{\max} = \max_{(u,v)\in E} t(u,v)$. Algorithm TRAMP thus avoids a potential $t^2_{\max}$ multiplicative factor. It is straightforward to verify, however, that Algorithm TRAMP computes the minimum cycle mean of G_{eqv}, and thus, the correctness of the algorithm follows immediately.

Line 24 of Algorithm TRAMP uses a shortest-paths subroutine that can either scale edge costs or not. The asymptotically fastest shortest-paths subroutine that does not scale edge costs is the $O(VE)$-time Bellman-Ford algorithm, which is used in the implementation shown in Figure 1. In this version of the algorithm $n = |V|$ initially, and line 10 is repeated $1 + \lfloor \lg(t^*/V) \rfloor$ times. (The parameter n in our algorithm eventually need only be greater than t^*, since the source vertex s introduced is connected

[2]Karp and Orlin [7] have shown that an extension of this formulation can be used to solve a generalization of the standard shortest-paths problem in which edge weights have a parameter λ subtracted from them. See also [11].

directly to all the vertices of G.) The ith iteration of lines 11–26 requires $O(VE + 2^i VE)$ steps. Therefore, the total running time is

$$\begin{aligned} O\left(\sum_{i=1}^{1+\lfloor \lg(t^*/V)\rfloor} (VE + 2^i VE)\right) &\leq O\left(2 \sum_{i=1}^{1+\lfloor \lg(t^*/V)\rfloor} 2^i VE\right) \\ &= O(VE + t^* E)\,. \end{aligned}$$

If we choose a shortest-paths subroutine in line 24 that scales costs, then the running time of the algorithm becomes dependent on the costs of edges. The asymptotically most efficient shortest-paths algorithms to date are those due to Gabow and Tarjan [3] and to Orlin and Ahuja [10] which run in $O(V^{1/2}E \lg(V c_{\max} t_{\max}))$ time. In such a scaling version of algorithm TRAMP, it is advantageous to let the initial value of n be $V^{1/2} \lg(V c_{\max} t_{\max})$. The number of iterations of lines 11–26 is $O(\lg(t^*/V^{1/2} \lg(V c_{\max} t_{\max})))$, and the overall algorithm terminates in $O(V^{1/2}E \lg(V c_{\max} t_{\max}) + t^* E)$ steps.

Acknowledgments

This research was supported in part by the Defense Advanced Research Projects Agency under Grant N00014-91-J-1698. We thank Jim Orlin of MIT for several helpful discussions, and Mark Hartmann of University of North Carolina at Chapel Hill for his detailed comments on an earlier draft of this paper.

References

[1] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* McGraw-Hill, MIT Press, 1990.

[2] G. B. Dantzig, W. O. Blattner, and M. R. Rao. Finding a cycle in a graph with minimum cost to time ratio with application to a ship routing problem. *Theory of Graphs*, 1967. P. Rosenstiehl, editor, Dunod, Paris, and Gordon and Breach, New York, pp. 77–84.

[3] H. N. Gabow and R. E. Tarjan. Faster scaling algorithms for network problems. *SIAM J. Computing*, October 1989.

[4] M. Hartmann. On cycle means and cyclic staffing. Technical Report UNC/OR/TR-90/14, University of North Carolina at Chapel Hill, June 1990.

[5] A. T. Ishii, C. E. Leiserson, and M. C. Papaefthymiou. Optimizing two-phase, level-clocked circuitry. *Advanced Research in VLSI and Parallel Systems: Proc. of the 1992 Brown/MIT Conference*, MIT Press, March 1992. To appear.

[6] R. M. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics*, 23:309–311, 1978.

[7] R. M. Karp and J. B. Orlin. Parametric shortest path algorithms with an application to cyclic staffing. *Discrete Applied Mathematics*, 3:37–45, 1981.

[8] E. L. Lawler. Optimal cycles in doubly weighted directed linear graphs. *Theory of Graphs*, 1967. P. Rosenstiehl, editor, Dunod, Paris, and Gordon and Breach, New York, pp. 209–214.

[9] E. L. Lawler. *Combinatorial Optimization, Networks and Matroids.* Holt, Rinehart and Winston, New York, 1976.

[10] J. B. Orlin and R. K. Ahuja. New scaling algorithms for the assignment and minimum cycle mean problem. Technical Report 2019-88, MIT Sloan School of Management, 1988.

[11] N. E. Young, R. E. Tarjan, and J. B. Orlin. Faster parametric shortest path and minimum balance algorithms. Technical Report 3112-90-MS, MIT Sloan School of Management, January 1990.