IMPROVED PRIMAL SIMPLEX ALGORITHMS FOR SHORTEST PATH, ASSIGNMENT AND MINIMUM COST FLOW PROBLEMS; SLOAN W. P. NO. 2090-88, NOVEMBER, 1988

Published @ 2017 Trieste Publishing Pty Ltd

ISBN 9780649739233

Improved Primal Simplex Algorithms for Shortest Path, Assignment and Minimum Cost Flow Problems; Sloan W. P. No. 2090-88, November, 1988 by Ravindra K. Ahuja & James B. Orlin

Edited by Trieste Publishing Pty Ltd.
Cover @ 2017

www.triestepublishing.com

RAVINDRA K. AHUJA & JAMES B. ORLIN

IMPROVED PRIMAL SIMPLEX ALGORITHMS FOR SHORTEST PATH, ASSIGNMENT AND MINIMUM COST FLOW PROBLEMS; SLOAN W. P. NO. 2090-88, NOVEMBER, 1988

Trieste

IMPROVED PRIMAL SIMPLEX ALGORITHMS FOR SHORTEST PATH, ASSIGNMENT AND MINIMUM COST FLOW PROBLEMS

Ravindra K. Ahuja
and
James B. Orlin

Sloan W.P. No. 2090-88 November, 1988

Improved Primal Simplex Algorithms for Shortest Path, Assignment and Minimum Cost Flow Problems

Abstract

In this paper, we present a new primal simplex pivot rule and analyse the worst-case complexity of the resulting simplex algorithm for the minium cost flow problem, the assignment problem and the shortest path problem. We consider networks with n nodes, m arcs, integral arc capacities bounded by an integer number U, and integral arc costs bounded by an integer number C. Let L and U denote the nonbasic arcs at their lower and upper bounds respectively, and $\bar{c}_{ij}$ denote the reduced cost of any arc (i, j). Further, let Δ be a parameter whose initial value is C. Then our pivot rule is as follows: Select as an entering arc any $(i, j) \in L$ with $\bar{c}_{ij} \leq -\Delta/2$ or any $(i, j) \in U$ with $\bar{c}_{ij} \geq \Delta/2$; select the leaving arc so that the strong feasibility of the basis is maintained. When there is no nonbasic arc satisfying this rule then replace Δ by $\Delta/2$. We show that the simplex algorithm using this rule performs O(nm U logC) pivots and can be implemented to run in $O(m^2 U \log C)$ time. Specializing these results for the assignment and shortest path problems we show that the simplex algorithm solves these problems in $O(n^2 \log C)$ pivots and O(nm logC) time. These algorithms use the same data structures that are typically used to implement the primal simplex algorithms for network problems and have enough flexibility for fine tuning the algorithms in practice. We also use these ideas to obtain an O(nm logC) label correcting algorithm for the shortest path problem with arbitrary arc lengths, and an improved implementation of Dantzig's pivot rule.

Subject Classification.

Networks/Graphs : Improved simplex algorithms for network flow problems

Keywords : Network flow algorithms, Minimum cost flow problem, Assignment problem, Shortest path problem, Simplex algorithm, Scaling, Matchings.

In this paper, we present a new primal simplex pivot rule and analyze the worst-case behavior of the resulting simplex algorithm for the minimum cost flow problem. These results are also specialized for the assignment and shortest path problems.

We consider a directed network $G = (N, A)$ with node set N and arc set A. Each arc $(i, j) \in A$ has a non-negative integer capacity u_{ij} and an integer cost c_{ij}. We denote by n, m, U, and C, the number of nodes, number of arcs, maximum arc capacity, and maximum absolute value of an arc cost, respectively. We represent by A(i) the set of arcs incident on node i, i.e., the set of incoming and outgoing arcs at node i. Let $deg(i) = |A(i)|$ denote the degree of node i for each $i \in N$.

The primal simplex algorithm for the minimum cost flow problem, subsequently referred to as the *network simplex algorithm*, has been extensively studied in the literature. Researchers have also been interested in developing polynomial time network simplex algorithms for the minimum cost flow problem and its special cases. Polynomial time primal simplex algorithms have been developed for the shortest path problem, the assignment problem and the maximum flow problem. The only polynomial time simplex algorithm for the minimum cost flow problem is a dual simplex algorithm due to Orlin [1984] which performs $O(n^3 \log n)$ pivots for the uncapacitated minimum cost flow problem. Developing a polynomial time primal simplex algorithm for the minimum cost flow problem is still an open problem.

We now briefly review the literature devoted to this area of research. Dial, Glover, Karney and Klingman [1979] and Zadeh [1979] showed that Dantzig's pivot rule (i.e., pivoting in the arc with minimum reduced cost) for the shortest path problem with non-negative arc lengths performs O(n) pivots when started from an all artificial basis. Roohy-Laleh [1980] in his unpublished Ph.D. thesis developed an alternative simplex pivot rule for the assignment problem for which the number of pivots is $O(n^3)$. Hung [1983] developed a simplex method for the assignment problem in which the number of pivots is $O(n^3 \log(nC))$.

Orlin [1985] showed that for integer data the primal simplex algorithm maintaining strongly feasible bases performs O(nm CU) pivots for any arbitrary pivot rule and O(nm U log (mCU)) for Dantzig's pivot rule. (This result was independently developed by Dantzig [1983].) When specialized to the shortest path problem with

arbitrary arc lengths and the assignment problem, the algorithm performs $O(n^2 \log(nC))$ pivots.

Akgul [1985a, 1985b] developed primal simplex algorithms for the shortest path and assignment problems that perform $O(n^2)$ pivots. Using simple data structures, Akgul's algorithms run in $O(n^3)$ time using simple data structure, and this time can be reduced to $O(nm + n^2 \log n)$ using Fibonacci heap data structure due to Fredman and Tarjan [1984]. Goldfarb, Hao and Kai [1986] describe another primal simplex algorithm for the shortest path problem whose number of pivots and running times are comparable to that of Akgul's algorithm.

Among the polynomial time dual simplex algorithms are the algorithms by Roohy-Laleh [1980] for the shortest path problem, by Balinski [1985] and Goldfarb [1985] for the assignment problem, and by Orlin [1984] for the minimum cost flow problem.

Recently, Goldfarb and Hao [1988] developed a polynomial time primal simplex algorithm for the maximum flow problem. This algorithm performs $O(nm)$ pivots and can be implemented to run in $O(n^2m)$ time. Tarjan [1988] showed how to use dynamic trees to further improve the running time to $O(nm \log n)$

This paper is based on the results contained in Orlin [1985] where the worst-case behavior of the primal simplex algorithm with Dantzig's pivot rule is analyzed. Our rule is essentially obtained by incorporating a scaling technique in Dantzig's pivot rule. Let L and U denote the nonbasic arcs at their lower and upper bounds respectively, and $\bar{c}_{ij}$ denote the reduced cost of any arc (i, j). Further, let Δ be a parameter whose initial value is C. Then our pivot rule is as follows: Select as an entering arc any $(i, j) \in L$ with $\bar{c}_{ij} \leq -\Delta/2$ or any $(i,j) \in U$ with $\bar{c}_{ij} \geq \Delta/2$; select the leaving arc so that the strong feasibility of the basis is maintained. (We discuss strong feasibility in detail in Section 1.1.) When there is no nonbasic arc satisfying this rule then replace Δ by $\Delta/2$. We call this pivot rule the *scaling pivot rule* and the simplex algorithm using the scaling pivot rule the *scaling network simplex algorithm.*

We show that the scaling network simplex algorithm solves the minimum cost flow problem in $O(nm \log C)$ pivots and can be implemented to run in $O(m^2 \log C)$ time. Specializing these results for the assignment and shortest path problems, we show that the scaling network simplex algorithm solves both of these problems in $O(n^2 \log C)$ pivots and $O(nm \log C)$ time. These results on the number of pivots are comparable to

that of Orlin [1985] for Dantzig's pivot rule, but the running times here are better by a factor of n.

Intuitively, the reason why the scaling network simplex algorithm is good is that the algorithm selects an entering arc with "sufficiently large" violation of the optimality conditions. This causes a "sufficiently large" improvement in the objective function and helps to show that the number of pivots are "sufficiently small." Further, the arc with "sufficiently large" violation can be picked up with little effort.

Among other results we obtain a scaling version of the label correcting algorithm that solves the shortest path problem with arbitrary arc lengths in O(nm logC) time. We also show that the primal simplex algorithm for the minimum cost flow problem with Dantzig's pivot rule can be implemented in $O(m^2 U \log C)$ heap operations where each heap operation takes O(min (log n, loglog(nC)) time.

The results in this paper rely on properties of the network simplex algorithm, strongly feasible bases and the "perturbation technique". Though we have tried to make this paper self-contained, we refer the reader to the papers of Orlin [1985] and Ahuja, Magnanti and Orlin [1988] for a more thorough discussion on these topics.

1. The Minimum Cost Flow Problem

In this section, we describe a network simplex algorithm that solves the minimum cost flow problem in O(nmU logC) pivots and can be implemented in $O(m^2 U \log C)$ time. The analysis of this algorithm, as specialized for the assignment problem and the shortest path problem, is presented in Sections 2 and 3.

1.1 Background

We consider the following node-arc formulation of the minimum cost flow problem.

$$\text{Minimize} \quad \sum_{(i,j) \in A} c_{ij} x_{ij} \qquad \text{(1a)}$$

subject to

$$\sum_{\{j:\,(i,\,j)\,\in\,A\}} x_{ij} - \sum_{\{j:\,(j,\,i)\,\in\,A\}} x_{ji} = b(i), \text{ for all } i \in N, \tag{1b}$$

$$0 \le x_{ij} \le u_{ij}, \text{ for each } (i, j) \in A. \tag{1c}$$

In this formulation, if $b(i) > 0$ then node i is a *supply node* and if $b(i) < 0$ then node i is a *demand node*. We assume that $\sum_{i \in N} b(i) = 0$. We also assume that the minimum cost flow problem has a feasible solution. The feasibility of the minimum cost flow problem can be ascertained by solving a maximum flow problem.

The simplex algorithm maintains a basic feasible solution at each stage. A basic solution of the minimum cost flow problem is denoted by a triple **(B, L, U)**; **B** , **L** and **U** partition the arc set A. The set **B** denotes the set of *basic arcs*, i.e., arcs of a spanning tree, and **L** and **U** respectively denote the sets of *nonbasic arcs* at their lower and upper bounds. We refer to the triple (B, L, U) as a *basis structure*. A basis structure (B, L, U) is called *feasible* if by setting $x_{ij} = 0$ for each $(i, j) \in L$, and by setting $x_{ij} = u_{ij}$ for each $(i, j) \in U$, the problem has a feasible solution satisfying (1b) and (1c).

A dual solution to the minimum cost flow problem is a vector π of *node potentials* and a vector $\bar{c}$ of *reduced costs* defined as $\bar{c}_{ij} = c_{ij} - \pi(i) + \pi(j)$. Since one of the mass balance constraint in (1b) is redundant, we can set one node potential arbitrarily. We henceforth assume that $\pi(1) = 0$.

A feasible basis structure **(B, L, U)** is called an *optimum* basis structure if it is possible to obtain a set of node potentials π so that the reduced costs satisfy the following optimality conditions:

$$\bar{c}_{ij} = 0, \text{ for each } (i, j) \in B, \tag{2a}$$

$$\bar{c}_{ij} \ge 0, \text{ for each } (i, j) \in L, \tag{2b}$$

$$\bar{c}_{ij} \le 0, \text{ for each } (i, j) \in U, \tag{2c}$$

Given a basis structure, the node potentials can be uniquely determined by setting $\pi(1) = 0$ and then using the (n-1) equations of (2a). A nonbasic arc (i, j) not satisfying (2b) or (2c), whichever is applicable, violates its optimality condition. In such a case, $|\bar{c}_{ij}|$ is called the *violation* of the arc (i, j).