IDENTIFYING THE OPTIMAL FACE OF A NETWORK LINEAR PROGRAM WITH A GLOBALLY CONVERGENT INTERIOR POINT METHOD*

MAURICIO G.C. RESENDE†, TAKASHI TSUCHIYA‡, AND GERALDO VEIGA§

Abstract. Based on recent convergence results for the affine scaling algorithm for linear programming, we investigate strategies to identify the optimal face of a minimum cost network flow problem. In the computational experiments described, one of the proposed optimality indicators is used to implement an early stopping criterion in DLNET, an implementation of the dual affine scaling algorithm for solving minimum cost network flow problems. We conclude from the experiments that the new indicator is far more robust than the one used in earlier versions of DLNET.

Key words. Linear programming, minimum cost network flow, indicator, affine scaling algorithm, computer implementation.

AMS(MOS) subject classifications. 65-05, 65F10, 65K05, 65Y05, 90C05, 90C06, 90C35

1. Introduction.

The dual affine scaling (DAS) algorithm [3] has been shown to perform well in practice on linear programming problems [1, 2, 7, 8], large-scale network flow problems [13], and large-scale assignment problems [11, 12]. In spite of its practical success, no polynomial time proof has been given for the primal or dual variants of the algorithm. The exception is the primal-dual affine scaling algorithm, for which a polynomial time proof exists [9], but that uses very short steps, rendering it impractical.

Recently, several authors (e.g. Dikin [4], Tsuchiya [16, 17], Tsuchiya & Muramatsu [19], Monteiro, Tsuchiya & Wang [10], Tsuchiya & Monteiro [18], Saigal [14] and Hall & Vanderbei [6]) have presented exciting convergence results for the affine scaling algorithm. In this paper, we use some results in [4, 10, 19] to derive indicators that identify the optimal primal and dual faces for a linear program, and present an approach for using one such indicator in the context of minimum cost network flow problems solved via the dual affine scaling algorithm. At the same time, we study some practical considerations of implementing the Dikin-Tsuchiya-Muramatsu step size strategy.

In Section 2 we state the dual affine scaling algorithm and the new convergence results. In Section 3 we briefly describe DLNET, the implementation of the dual affine scaling algorithm used in the computational study. A stopping strategy based on the solution of a maximum flow problem is outlined in Section 4. In Section 5, we discuss two indicators implemented in DLNET. Computational results are described in Section 6 and concluding remarks are made in Section 7.

The following notation is used throughout our paper. We denote the vector of all ones by e. Its dimension is always clear from the context. R^n, R^n_+ and R^n_{++} denote the n-dimensional Euclidean space, the nonnegative orthant of R^n and the positive orthant of R^n, respectively. The set of all $m \times n$ matrices with real entries is denoted by $\mathsf{R}^{m\times n}$. Given an index set $J \subseteq \{1, \ldots, n\}$ and a vector $w \in \mathsf{R}^n$, we denote by w_J the subvector of w corresponding to J. Similarly, if E is an $m \times n$ matrix then E_J denotes the $m \times |J|$ submatrix of E corresponding to J. The Euclidean norm, the 1-norm and the ∞-norm are denoted by $\|\cdot\|$, $\|\cdot\|_1$ and $\|\cdot\|_\infty$, respectively. If J is a finite index set then $|J|$ denotes its cardinality, that is the number of elements of J. The superscript $^\top$ denotes transpose.

* April 1993 (Revised August 1993). To appear in *Large Scale Optimization: State of the Art*, W.W. Hager, D.W. Hearn, and P.M. Pardalos, eds., Kluwer Academic Publishers B.V. (1993)

† AT&T Bell Laboratories, Murray Hill, NJ 07974 USA

‡ The Institute of Statistical Mathematics, Tokyo, 106, Japan

§ Department of IEOR, University of California, Berkeley,CA 94720 USA

2. Convergence results and new indicators. Let $A \in \mathsf{R}^{m\times n}$, $c, x, s \in \mathsf{R}^n$ and $b, y \in \mathsf{R}^m$. The dual affine scaling algorithm solves the linear programming problem

$$\begin{aligned} &\text{minimize} \quad c^\top x \\ &\text{subject to} \quad Ax = b, \quad x \geq 0 \end{aligned} \tag{2.1}$$

indirectly by solving its dual

$$\begin{aligned} &\text{maximize} \quad b^\top y \\ &\text{subject to} \quad A^\top y + s = c, \quad s \geq 0. \end{aligned} \tag{2.2}$$

The algorithm starts with an initial solution

$$y^0 \in \{y \mid s = c - A^\top y > 0\} \tag{2.3}$$

and obtains iterate y^{k+1} from y^k according to

$$y^{k+1} = y^k + \alpha^k d_y^k, \tag{2.4}$$

where the search direction d_y is

$$d_y^k = (AS_k^{-2}A^\top)^{-1}b \tag{2.5}$$

and

$$S_k = \mathrm{diag}(s_1^k, ..., s_n^k). \tag{2.6}$$

We take a step moving a fraction γ $(0 < \gamma < 1)$ of the way to the boundary of the feasible region at each iteration, namely,

$$\alpha^k = \gamma \times \min\{-s_i^k/(d_s^k)_i \mid (d_s^k)_i < 0,\ i = 1, ..., n\}, \tag{2.7}$$

where $d_s^k = -A^\top d_y^k$ is a unit displacement vector in the space of slack variables. At each iteration, a tentative primal solution is computed by

$$x^k = S_k^{-2}A^\top(AS_k^{-2}A^\top)^{-1}b. \tag{2.8}$$

It is easy to check $Ax^k = b$, but x^k may not necessarily be positive. The set of optimal solutions is referred to as the *optimal face*. We use the index set N_* for the always-active index set on the optimal face of the primal, and B_* for its complement. It is well-known that B_* is the always-active index set on the optimal face of the dual, and N_* is its complement. An indicator is a quantity to detect whether an index belongs to N_* or B_*. In the remaining part of this section we propose several indicators developed from the convergence theory of the affine scaling algorithm [4, 10, 19].

We start with the following basic result, which states, under a very weak condition, that the iterative sequence of the algorithm converges to a relative interior point of a face on which the objective function is constant.

THEOREM 2.1. *(Lemma 1.2 of [19], Theorem 2.6 of [10]) The sequence $\{y^k\}$ converges to an interior point of a face on which the objective function is constant. Let B be the always-active index set on the face and N be its complement, and let b^∞ be the limiting objective function value. Then we have a constant $C_0 > 0$ such that*

$$\limsup_{k\to\infty} \frac{s_i^k}{b^\infty - b^\top y^k} \leq C_0 \tag{2.9}$$

for all $i \in B$, while

$$\frac{s_i^k}{b^\infty - b^\top y^k} \tag{2.10}$$

diverges to infinity for all $i \in N$. ■

We denote by s^∞ the limiting slack vector. We have $s_N^\infty > 0$ and $s_B^\infty = 0$. First we consider indicators to detect N and B. The following vector plays an important role:

$$u^k \equiv \frac{(S^k)^{-1} d_s^k}{b^\infty - b^\top y^k} = \frac{S^k x^k}{b^\infty - b^\top y^k}. \tag{2.11}$$

LEMMA 2.2. *We have*

$$\lim_{k\to\infty} (u^k)^\top e = \lim_{k\to\infty} \frac{(s^k)^\top x^k}{b^\infty - b^\top y^k} = 1. \quad \blacksquare \tag{2.12}$$

The lemma is obtained as a direct consequence of Lemma 3.8 of [10]. The use of this lemma is that we can estimate $b^\infty - b^\top y^k$ by $(s^k)^\top x^k$ asymptotically. Consequently, (2.9) can be stated as

$$\lim_{k\to\infty} \sup \frac{s_i^k}{(s^k)^\top x^k} \le C_0.$$

Then, if $i \in B$, for any β such that $0 < \beta < 1$, we have that

$$\lim_{k\to\infty} \sup \frac{s_i^k}{((s^k)^\top x^k)^\beta} \to 0,$$

since $((s^k)^\top x^k)^\beta$ converges to zero at a slower rate than $((s^k)^\top x^k)$ for any β such that $0 < \beta < 1$. Therefore, if we chose $\beta = 1/2$, we have following indicator:

Indicator 1: Let $C_1 > 0$ be any constant, and define index set N^k as the index set consisting of the indices i for which

$$s_i^k \le C_1 \sqrt{(s^k)^\top x^k}. \tag{2.13}$$

Then $N^k = N_*$ holds asymptotically. ■

This indicator is available under the weak assumptions of Theorem 2.1, so that it can be used to detect B_* and N_* without any substantial restriction on step-size. On the other hand, it gives the correct partition only if the limit point y^∞ happens to be a relative interior point of the optimal face of the dual (since Theorem 2.1 does not guarantee global convergence) and thus lacks a theoretical justification. However, since we know by experience that y^∞ usually lies in the relative interior of the optimal face, we may expect that it should work well in practice. Another potential problem with this indicator is that it is not scaling invariant, so that it will behave differently if the scaling of the problem is changed.

Now we assume that the step-size is asymptotically less than or equal to 2/3. Then the limiting point exists in the interior of the optimal face and b^∞ is the optimal value. Specifically, we have the following theorem.

THEOREM 2.3. *(Theorem 1.1. of [19] (see also Theorem 3.1, Theorem 4.2 and Theorem 4.3 of [10]).) If $\gamma \le 2/3$ throughout the iterations, then*

- *$\{y^k\}$ converges to an interior point of the optimal face of the dual problem.*
- *$\{x^k\}$ converges to the analytic center of the optimal face of the primal problem.*
- *$\{b^\top y^k\}$ converges linearly to the optimal value b^∞ asymptotically, where the (asymptotic) reduction rate is exactly $1 - \gamma$.*

The following theorem can be used to define indicators.

THEOREM 2.4. *(Lemma 4.1 of [10])*

$$\lim_{k\to\infty} u_i^k \to 1/|B_*| \quad for \quad i \in B_* \tag{2.14}$$

$$\lim_{k\to\infty} u_i^k \to 0 \quad otherwise. \ \blacksquare \tag{2.15}$$

The vector u^k is not available because we do not have the exact optimal value, but we can estimate $b^\infty - b^\top y^k$ by $(s^k)^\top x^k$ to obtain

$$\lim_{k\to\infty} \frac{s_i^k x_i^k}{(s^k)^\top x^k} \to 1/|B_*| \quad for \quad i \in B_* \tag{2.16}$$

$$\lim_{k\to\infty} \frac{s_i^k x_i^k}{(s^k)^\top x^k} \to 0 \quad otherwise. \ \blacksquare \tag{2.17}$$

On the basis of this fact, we propose the following procedure to construct N^k which asymptotically coincides with B_*:

Indicator 2: Let δ be a constant between 0 and 1. We obtain N^k according to the following procedure:

- Step 1: Sort $h_i^k = s_i^k x_i^k/(s^k)^\top x^k$ according to its order of magnitude. Here we denote i_l the index for the l-th largest component.
- Step 2: For $p := 1, 2, ...$ compare h_{i_p} and δ/p, and let p^* be the first number such that $h_{i_p^*} \le \delta/p^*$. Then set

$$N^k = \{i_1, i_2, ..., i_{p^*-1}\}. \quad \blacksquare \tag{2.18}$$

Now, we turn our attention to asymptotic behavior of s_i^{k+1}/s_i^k. If $i \in N_*$, then s_i^k converges to a positive value, and hence

$$\lim_{k\to\infty} \frac{s_i^{k+1}}{s_i^k} = 1. \tag{2.19}$$

If $i \in B_*$, s_i^k converges to zero. Recall Theorem 2.4 which states

$$\lim_{k\to\infty} \frac{s_i^k x_i^k}{b^\infty - b^\top y^k} = \frac{1}{|B_*|}. \tag{2.20}$$

Since x_i^k converges to a positive number (Theorem 2.3(ii)) and the objective function reduces with a rate of $1-\gamma$ (Theorem 2.3(iii)), then

$$\lim_{k\to\infty} \frac{s_i^{k+1}}{s_i^k} = 1 - \gamma \tag{2.21}$$

holds. Thus we are naturally lead to the following indicator:

Indicator 3: Take a constant η such that $1-\gamma < \eta < 1$. Then let

$$N^k = \{i \mid \frac{s_i^{k+1}}{s_i^k} \ge \eta\} \tag{2.22}$$

be defined as the index set. Then $N^k = N_*$ holds asymptotically. ■

Of the three indicators described here, Indicators 2 and 3 stand on the firmest theoretical basis. Furthermore, unlike Indicator 1, both are scaling invariant. Indicator 3 seems to be easier to implement. In the following sections we discuss the implementation and testing of Indicator 3 to identify the optimal primal-dual structure of minimum cost network flow problems solved via the dual affine scaling algorithm.

3. The implementation. In this section, we review DLNET, the implementation of the DAS algorithm for minimum cost network flow problems used in this study. For a detailed description of DLNET, we refer the reader to [13]. We begin by stating the minimum cost network flow (MCNF) problem which DLNET is designed to solve.

Consider a network with an underlying directed graph $G = (V, E)$, where V is a set of m vertices and E a set of n edges. Let (i, j) denote a directed edge from vertex i to vertex j. For each vertex $i \in V$, let b_i denote the net flow out of vertex i. If $b_i > 0$ vertex i is a source, if $b_i < 0$ vertex i is a sink and, otherwise, vertex i is a transshipment vertex. For each edge $(i, j) \in E$, let c_{ij}, l_{ij} and u_{ij} denote, respectively, the unit flow cost, lower bound and upper bound on flow in edge (i, j). All data are assumed to be integer. A feasible solution of a network flow problem (often referred to as *flow*) is given by the n-dimensional vector x, where component x_{ij} is the flow in edge (i, j), satisfying flow conservation constraints for all vertices and flow lower bound and capacity constraints on all edges.

The minimum cost network flow (MCNF) problem consists of finding a flow of minimum cost, as expressed in the following classical linear programming formulation:

$$\min \sum_{ij \in E} c_{ij} x_{ij} \tag{3.1}$$

subject to:

$$\sum_{jk \in E} x_{jk} - \sum_{kj \in E} x_{kj} = b_j, \quad j \in V \tag{3.2}$$

$$l_{ij} \leq x_{ij} \leq u_{ij}, \quad (i, j) \in E. \tag{3.3}$$

More compactly, the linear program in (3.1-3.3) can be expressed as

$$\min \{c^\top x \mid Ax = b,\ l \leq x \leq u\},$$

where A is the incidence matrix of G. We denote the i-th column of A by A_i, the i-th row of A by $A_{\cdot i}$ and a submatrix of A formed by columns with indices in set S by A_S. If graph G has p connected components, there are exactly p redundant flow conservation constraints, which are sometimes removed from the problem formulation. We rule out a trivially infeasible problem by assuming

$$\sum_{j \in V^k} b_j = 0, \quad k = 1, \ldots, p, \tag{3.4}$$

where V^k is the set of vertices for the k-th component of G.

Often, it is further required that x_{ij} be integer, i.e. we replace (3.3) with

$$l_{ij} \leq x_{ij} \leq u_{ij},\ x_{ij} \text{ integer}, \ (i, j) \in E. \tag{3.5}$$

In the remainder of this paper we assume, without loss of generality, that $l_{ij} = 0$ for all $(i, j) \in E$ and that $c \neq 0$.

The variant of the DAS algorithm, implemented in DLNET, solves the MCNF linear program

$$\min \{c^\top x \mid Ax = b,\ 0 \le x \le u\}.$$

DLNET does so indirectly by solving the dual program

$$\max \{b^\top y - u^\top z \mid A^\top y - z + s = c,\ z \ge 0, s \ge 0\} \tag{3.6}$$

where z and s are an n-dimensional vectors and y is an m-dimensional vector.

The steps of the algorithm are as follows:

- Step 0: Set $k = 0$ and let the initial interior point solution be given by:

$$\begin{aligned} y_i^k &= 0,\ i = 1, \ldots, n \\ s_i^k &= c_i + 2\|c\|,\ i = 1, \ldots, n \\ z_i^k &= 2\|c\|,\ i = 1, \ldots, n. \end{aligned}$$

- Step 1: Let $k = k + 1$, and solve

$$\begin{aligned} A(Z_k^2 + S_k^2)^{-1} A^\top \Delta y &= b - AZ_k^2 (Z_k^2 + S_k^2)^{-1} u, \\ \Delta z &= Z_k^2 (Z_k^2 + S_k^2)^{-1} (A^\top \Delta y - S_k^2 u), \\ \Delta s &= \Delta z - A^\top \Delta y, \end{aligned}$$

  for the ascent direction $\{\Delta y, \Delta z, \Delta s\}$ where

$$Z_k = \mathrm{diag}(z_1^k, \ldots, z_n^k) \text{ and } S_k = \mathrm{diag}(s_1^k, \ldots, s_n^k).$$

- Step 2: Compute the step size $\alpha = \gamma \times \min\{\alpha_z, \alpha_s\}$, where $0 < \gamma < 1$ and

$$\alpha_z = \min\{-z_i^k / (\Delta z)_i \mid (\Delta z)_i < 0,\ i = 1, \ldots, n\}$$

$$\alpha_s = \min\{-s_i^k / (\Delta s)_i \mid (\Delta s)_i < 0,\ i = 1, \ldots, n\}.$$

- Step 3: Move to the new iterate:

$$\{y^{k+1}, z^{k+1}, s^{k+1}\} = \{y^k, z^k, s^k\} + \alpha \{\Delta y, \Delta z, \Delta s\}.$$

- Step 4: Check optimality. If optimal primal-dual solution is found, then stop, else go to Step 1.

The bulk of the work in the DAS algorithm is related to building and updating the matrix $AD_kA^\top$ and solving the system of linear equations

$$AD_k A^\top \Delta y = b - AZ_k^2 D_k u, \tag{3.7}$$

where $D_k = (Z_k^2 + S_k^2)^{-1}$. DLNET relies heavily on a preconditioned conjugate gradient algorithm to solve the direction finding system at each iteration. The code uses two preconditioners: the standard diagonal preconditioner and a maximum weighted spanning tree preconditioner.

In the next section we describe in more detail how optimality checking has been implemented in DLNET.

4. Stopping with a boundary solution. The DAS algorithm generates a sequence of dual interior solutions, with a corresponding sequence of tentative primal solutions [15]. Under very mild conditions, these sequences converge, respectively, to the relative interiors of the primal and dual optimal faces [4, 19]. DLNET requires a stopping strategy that guarantees termination with a primal integer solution and uses MCNF-specific properties to stop the algorithm earlier than its theoretical convergence.

This study concentrates on a stopping strategy based on the solution of a maximum flow problem. As described in [13], at each iteration of the DAS algorithm, DLNET tries to identify a set of active edges defining the supporting affine space of the optimal dual face. According to an indicator function, the algorithm selects a tentative set of active edges $\mathcal{F}$.

Unless the DAS algorithm is close to convergence, there is no guarantee that $\mathcal{F}$ actually defines a dual face, as the set $\{y \in \Re^m \mid A_{\mathcal{F}}^\top y = c_{\mathcal{F}}\}$ can be empty. Instead, the supporting affine space is defined by a maximal forest $\mathcal{T}$ of graph $G_{\mathcal{F}} = (V, \mathcal{F})$. We select this maximal forest by computing maximum weighted spanning trees for each component of $G_{\mathcal{F}}$, using as weights the diagonal elements of the current scaling matrix. Whenever $\mathcal{F}$ defines a dual face, edges in $\mathcal{F} \setminus \mathcal{T}$ correspond to redundant hyperplanes, and the dual face is unique. A tentative dual optimal solution is computed by projecting the current dual interior vector y^k onto the supporting affine space of the dual face defined by $\mathcal{T}$,

$$\min_{y^* \in \Re^m} \{\|y^* - y^k\| \mid A_{\mathcal{T}}^\top y^* = c_{\mathcal{T}}\}.$$

The dual slacks for the tentative dual optimal solution are computed as

$$z_i^* = \begin{cases} -\delta_i & \text{if } \delta_i < 0 \\ 0 & \text{otherwise} \end{cases} \qquad s_i^* = \begin{cases} 0 & \text{if } \delta_i < 0 \\ \delta_i & \text{otherwise,} \end{cases}$$

where $\delta_i = c_i - A^\top y^*$.

Based on the projected dual solution y^*, we select a refined tentative optimal face by redefining the set of active edges as

$$\tilde{\mathcal{F}} = \{i \in E \mid |c_i - A_{\cdot i}^\top y^*| < \epsilon\}.$$

Next, we attempt to build a primal feasible solution, x^*, complementary to the tentative dual optimal solution by setting the inactive edges to lower or upper bounds, i.e., for $i \in E \setminus \tilde{\mathcal{F}}$,

$$x_i^* = \begin{cases} 0 & \text{if } i \in \Omega^+ = \{i \in E \setminus \tilde{\mathcal{F}} \mid c_i - A_{\cdot i}^\top y^* > 0\} \\ u_i & \text{if } i \in \Omega^- = \{i \in E \setminus \tilde{\mathcal{F}} \mid c_i - A_{\cdot i}^\top y^* < 0\}. \end{cases}$$

By considering only the active edges, we build a *restricted network*, represented by the constraint set

$$A_{\tilde{\mathcal{F}}} x_{\tilde{\mathcal{F}}} = \tilde{b} = b - \sum_{i \in \Omega^-} u_i A_i, \tag{4.1}$$

$$0 \le x_i \le u_i, \quad i \in \tilde{\mathcal{F}}. \tag{4.2}$$

Clearly, from the flow balance constraints (4.1), if a feasible flow $x_{\tilde{\mathcal{F}}}^*$ for the restricted network exists, it defines, along with $x_{\Omega^+}^*$ and $x_{\Omega^-}^*$, a primal feasible solution complementary to y^*. A feasible flow for the restricted network can be determined by solving a maximum flow problem on the *augmented network* defined by underlying graph $\tilde{G} = (\tilde{V}, \tilde{E})$, where

$$\tilde{V} = \{\sigma\} \cup \{\theta\} \cup V$$

and

$$\tilde{E} = \Sigma \cup \Theta \cup \tilde{\mathcal{F}}.$$

In addition, for each edge $(i, j) \in \tilde{\mathcal{F}}$ there is an associated capacity u_{ij}. The additional edges are such that

$$\Sigma = \{(\sigma, i) \mid i \in V^+\},$$

with associated capacity $\tilde{b}_i$ for each edge (σ, i), and

$$\Theta = \{(i, \theta) \mid i \in V^-\},$$

with associated capacity $-\tilde{b}_i$ for each edge (i, θ), where $V^+ = \{i \in V \mid \tilde{b}_i > 0\}$ and $V^- = \{i \in V \mid \tilde{b}_i < 0\}$.

From the proposition below, proved in [13], we conclude that finding a feasible flow for the restricted network involves the solution of a maximum flow problem. Furthermore, this feasible flow is integer, as we can select a maximum flow algorithm that provides an integer solution.

PROPOSITION 1. *Let $\mathcal{M}_{\sigma,\theta}$ be the maximum flow value from σ to θ, and $\tilde{x}$ a maximal flow on the augmented network. Then, $\mathcal{M}_{\sigma,\theta} = \sum_{i \in V^+} \tilde{b}_i$ iff $\tilde{x}_{\tilde{\mathcal{F}}}$ is a feasible flow for the restricted network.*

5. Indicators used in DLNET. In the most recent version of DLNET, users can choose between two indicator functions used by the program to identify an optimal face. In addition to the Dual Slack indicator used in the experiments described in [13]), Indicator 3, presented in Section 2, is also available. In this section, we compare these indicators illustrating some of their properties with test runs performed on a a small MCNF problem with 256 vertices and 2048 edges.

5.1. Dual Slack Indicator. When using the dual slack indicator, a tentative dual optimal face is built by marking edge i as active if the corresponding dual slack variables z_i and s_i are either both very small or of the same order of magnitude, i.e.,

$$\mathcal{F} = \{i \in E \mid |s_i^k - z_i^k| < \epsilon_0 \text{ or } \epsilon_1 \leq s_i^k / z_i^k \leq 1/\epsilon_1\},$$

where $\epsilon_0 > 0$ and $1 > \epsilon_1 > 0$ are small tolerances. The remaining edges are assigned to either to lower or upper bound as describe in Section 4.

Figures 5.1-5.3 display, for DLNET running with different step factors, the behavior of the dual slack indicator performed at each iteration of the algorithm. The different step factors are considered for the purposes of comparing with Indicator 3, whose theoretical properties depend on the step size factor. From these plots, one can observe the effect of step size on the convergence of the indicator. For step factor $\gamma = .50$, the indicator correctly identifies the optimal face at iteration 43, for step factor $\gamma = .66$ at iteration 32, and for step factor $\gamma = .90$ at iteration 23. This is the expected behavior accountable to the faster progress at each iteration when longer steps are taken.

5.2. Indicator 3. With Indicator 3, DLNET attempts to identify the set of active edges defining the optimal face by examining the ratio between subsequent iterates of each dual slack. Let γ be the step size factor. From the convergence results presented in Section 2, for each edge $i \in E$, we have:

- Edge i is set at its upper bound if:

$$\lim_{k \to \infty} s_i^k / s_i^{k-1} = 1 - \gamma \quad \text{and} \quad \lim_{k \to \infty} z_i^k / z_i^{k-1} = 1.$$

- Edge i is set at its lower bound if:

$$\lim_{k \to \infty} s_i^k / s_i^{k-1} = 1 \quad \text{and} \quad \lim_{k \to \infty} z_i^k / z_i^{k-1} = 1 - \gamma.$$

- Edge i is left active if:

$$\lim_{k \to \infty} s_i^k / s_i^{k-1} = 1 - \gamma \text{ and } \lim_{k \to \infty} z_i^k / z_i^{k-1} = 1 - \gamma.$$

For a DLNET run using a fixed step factor $\gamma = .66$, Figures 5.5-5.6 display the typical behavior of the ratio between subsequent iterates for the slack variables when, at optimality, the corresponding edge is active, at the upper bound or at the lower bound.

Scale invariance is the most interesting feature of this indicator. An implementable version can be created by selecting constants which depend only on the step size factor γ. In DLNET, we take $\kappa_0 = .7$ and $\kappa_1 = .9$, and, at each iteration of the algorithm, classify edges in the following manner:

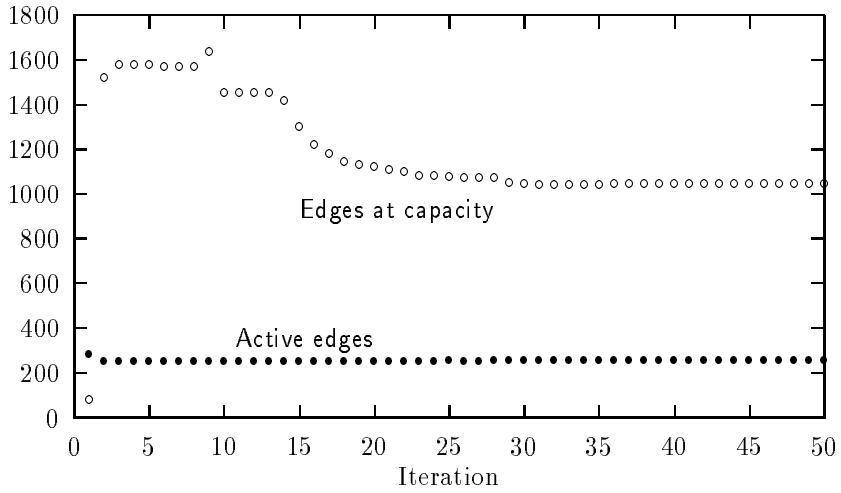

FIG. 5.1. *Dual Slack Indicator (Step factor $\gamma = .50$)*

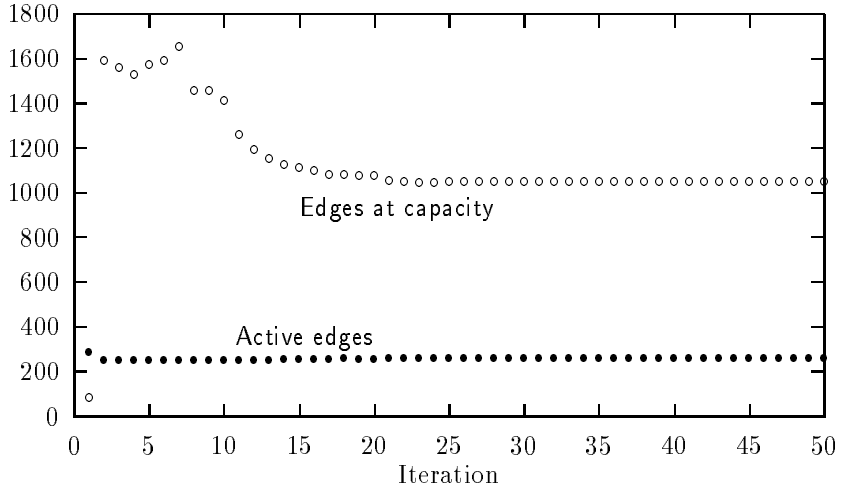

FIG. 5.2. *Dual Slack Indicator (Step factor $\gamma = .66$)*

- Edge i is set to upper bound if:

$$s_i^k / s_i^{k-1} < \kappa_0 \text{ and } z_i^k / z_i^{k-1} > \kappa_1.$$

- Edge i is set to lower bound if:

$$s_i^k / s_i^{k-1} > \kappa_1 \text{ and } z_i^k / z_i^{k-1} < \kappa_0.$$

- Otherwise, the edge i are left active, defining the tentative optimal face.

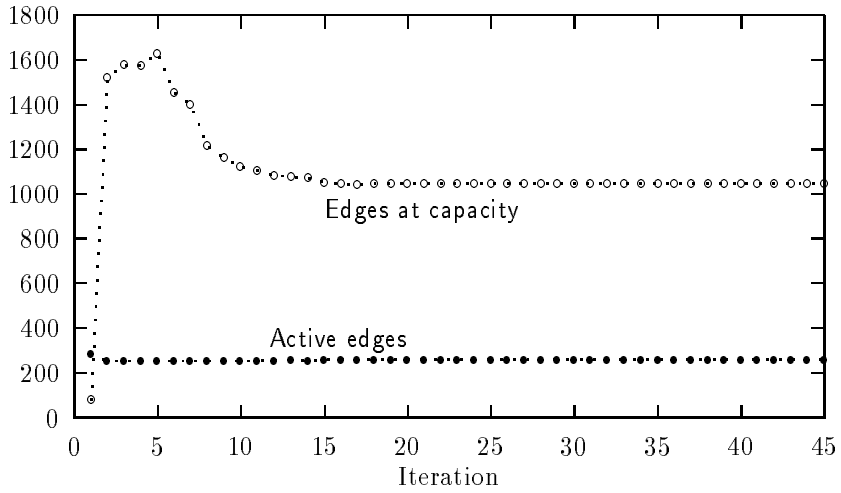

FIG. 5.3. *Dual Slack Indicator (Step factor $\gamma = .90$)*

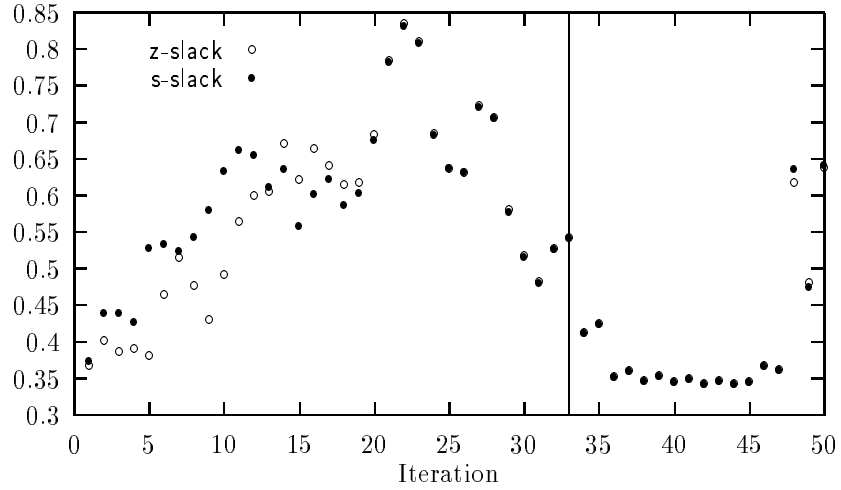

FIG. 5.4. *Dual Slack Ratio (Edge left active)*

Indicator 3 displays a behavior analogous to the dual slack indicator, as evidenced in Figures 5.7-5.9. For step factor $\gamma = .50$, the indicator correctly identifies the optimal face at iteration 45, for step factor $\gamma = .66$ at iteration 34, and for step factor $\gamma = .90$ at iteration 24. This last result is somewhat unexpected, as the convergence results do not hold for step factor $\gamma > \frac{2}{3}$. However, as the computational experiments reported in Section 6 show, if used as part of an early stopping strategy like the one implemented in DLNET, this indicator can be successfully used with large step factors.

6. Experimental results. In this section, we describe the results of a computational experiment with two indicators and two affine scaling step size rules. Our objective is to determine if the

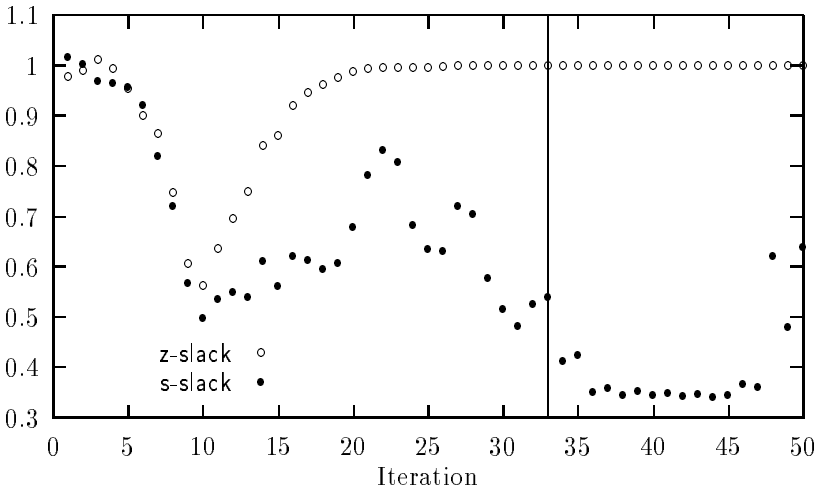

FIG. 5.5. *Dual Slack Ratio (Edge set to upper bound)*

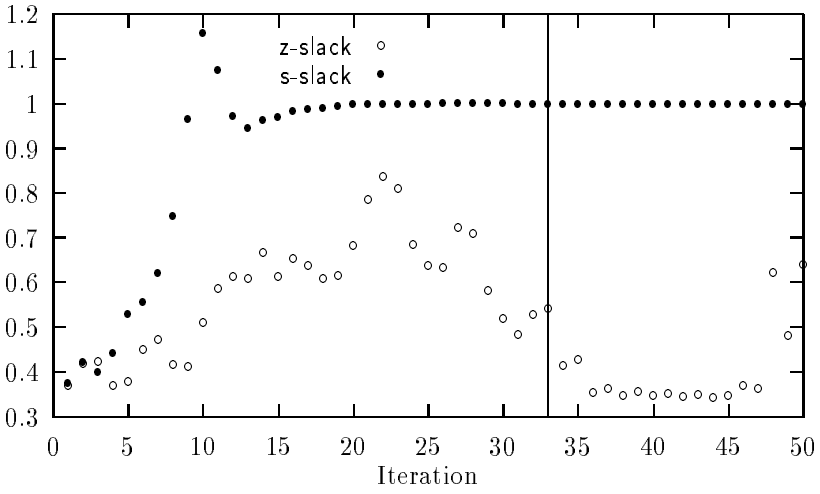

FIG. 5.6. *Dual Slack Ratio (Edge set to lower bound)*

new indicator is effective and the practical effect of a smaller step size in the affine scaling algorithm.

The indicators selected for this experiment are the dual slack indicator [13] and Indicator 3, described in Sections 2 and 4.

The first step size rule used was the "fixed step size," used in previous implementations of the DAS algorithm [2, 12, 13]. That rule dictates that the first 10 DAS iterations use a fixed step size parameter $\gamma = 0.99$ and thereafter 0.95. The second rule satisfies the requirements of the globally convergent DAS algorithm. The algorithm begins using a step size parameter $\gamma = 0.99$. When the

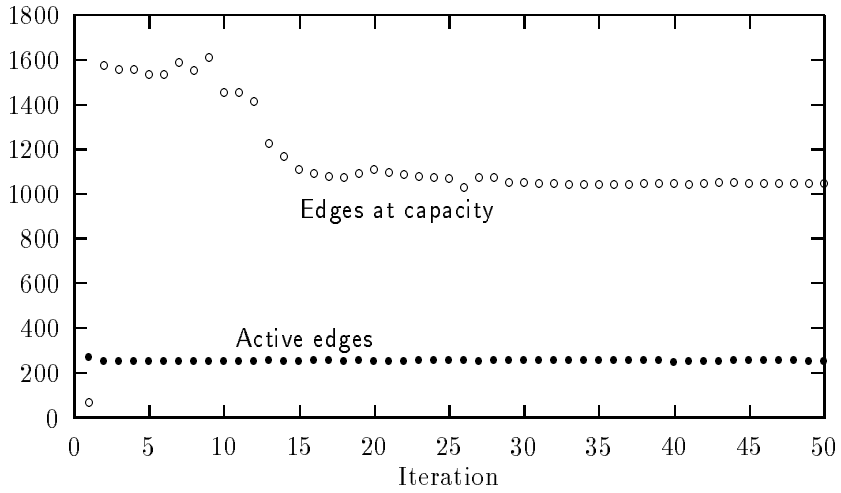

FIG. 5.7. *Indicator 3 (Step factor $\gamma = .50$)*

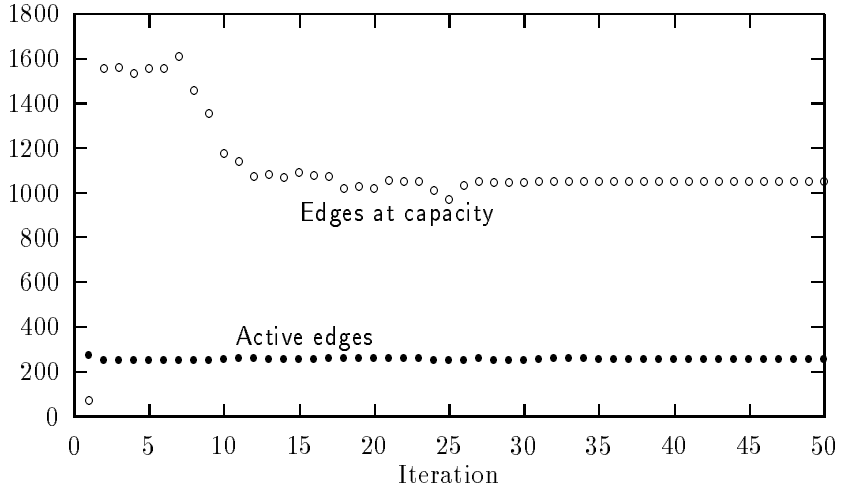

FIG. 5.8. *Indicator 3 (Step factor $\gamma = .66$)*

relative dual objective function improvement

$$\mathcal{I}^k = \frac{(b^\top y^k - u^\top z^k) - (b^\top y^{k-1} - u^\top z^{k-1})}{|b^\top y^k - u^\top z^k|}$$

falls below 0.1, the code begins using a parameter value $\gamma = 0.95$. When $\mathcal{I}^k < 0.01$, the code uses $\gamma = 0.5$ from then on.

To compare the four step/indicator combinations, DLNET was run with the maximum flow primal-dual optimality test. The primal-dual optimal solution is tested every five DAS iterations

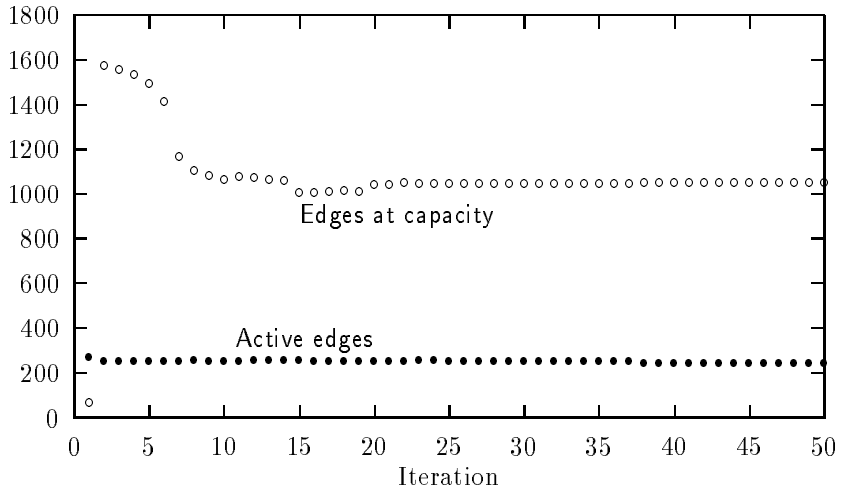

FIG. 5.9. *Indicator 3 (Step factor $\gamma = .90$)*

TABLE 6.1
Grid-Density-16 problem family – iterations

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
1024	16384	60	44	60	44
2048	32768	67	51	67	50
4096	65536	67	52	67	54
8192	131072	93	68	93	68
16384	262144	90	64	89	63
32768	524288	90	65	90	64

until $\mathcal{I}^k < 0.001$, when from that point on, testing is done every DAS iteration. This testing frequency is clearly not appropriate in practice, since a maximum flow problem eventually must be solved at every iteration of the interior point algorithm. Later in this section, we report for comparison, iterations and running times of DLNET using default parameter settings.

The computational experiments were conducted using DLNET version 2.2a (9 Dec 92) on a Silicon Graphics IRIS computer, model 4D/240, with four 25 MHz IP7 processors, a MIPS R2010A/R3010 FPU, a MIPS R2000A/R3000 CPU, 64 Kbytes instruction cache, 64 Kbytes data cache, 256Kbytes of secondary data cache, and 256 Mbytes of main memory. The operating system is IRIX System V Release 4.0.5. Each run of the code was done on a single processor.

The test problems used in the computational experiment were taken from the suite of test problems composed for the The First DIMACS International Algorithm Implementation Challenge [5]. All problem instances are generated with problem generators distributed for the challenge. The problem classes used are Grid-Density-16, Grid-Wide, Grid-Long, Mesh-1 and Netgen-Lo. All problem generators can be obtained via anonymous `ftp` from `dimacs.rutgers.edu`. Instances having approximately 1024, 2048, 4096, 8192, 16384 and 32768 vertices were generated for each problem class, except Mesh-1, for which instances having 256, 1024, 4096, 16384 and 65536 were generated. A single instance of each size was generated always using random number generator seed

TABLE 6.2
Grid-Density-16 problem family – CPU seconds

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
1024	16384	153.2	109.3	146.2	103.3
2048	32768	432.9	299.5	399.1	290.2
4096	65536	1004.6	758.1	1054.8	841.4
8192	131072	3764.2	2606.2	3600.3	2739.1
16384	262144	10069.9	6827.1	9454.3	6492.0
32768	524288	22034.0	15408.0	22537.7	14829.9

TABLE 6.3
Grid-Long problem family – iterations

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
1026	2000	Failed	Failed	31	29
2050	3984	Failed	Failed	45	37
4098	7952	Failed	Failed	60	50
8194	15888	Failed	Failed	84	61
16386	31760	Failed	Failed	91	81
32770	63504	Failed	Failed	124	101

270001. A description of the instances is given in [13].

The results are summarized in two tables for each problem family. The first table summarizes number of iterations for each indicator/step size rule pair while the second table summarizes CPU times, in seconds. An entry "Failed" indicates that the code failed to solve the problem, i.e. it failed to identify a primal-dual optimal integer solution in 200 DAS iterations. Tables 6.1 and 6.2 summarize the runs for problem family Grid-Density-16. Tables 6.3 and 6.4 summarize the runs for problem family Grid-Long. Tables 6.5 and 6.6 summarize the runs for problem family Grid-Wide. Tables 6.7 and 6.8 summarize the runs for problem family Mesh-1. Tables 6.9 and 6.10 summarize the runs for problem family Netgen-Lo.

We make the following remarks about the computational experiment:

- To be considered successful, the runs in this experiment required that integer primal-dual optimal solutions be produced. This differs from the experiments described in [13], where only integer primal optimal solutions were required.
- The new indicator was clearly better than the dual slack indicator used in earlier versions of DLNET. In all problem classes, except Grid-Density-16, the dual slack indicator failed in at least half of the instances. In two classes it failed for all instances. The new indicator did not fail a single time.
- In the single class (Grid-Density-16) in which the dual slack indicator was successful in identifying the primal-dual optimal solution, the number of iterations required by DLNET differed by at most 1, for a given step size rule. Running times were in favor of the variant with the new indicator in 8 of the 12 runs.
- The following observations are for runs using the new indicator. The number of iterations required by the dynamic step size approach increased with respect to the number of iterations required by the fixed step size approach from a low of 0% to a high of 46%. The average increases in number of iterations for classes Grid-Density-16, Grid-Long, Grid-Wide, Mesh-1 and Netgen-Lo, were respectively, 36%, 20%, 13%, 17% and 31%. Running times increased from a low of 5% to a high of 60%. The average increases in running time for classes Grid-Density-16, Grid-Long,

TABLE 6.4
Grid-Long problem family – CPU seconds

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
1026	2000	Failed	Failed	12.2	11.1
2050	3984	Failed	Failed	46.6	38.4
4098	7952	Failed	Failed	195.0	157.8
8194	15888	Failed	Failed	745.6	518.1
16386	31760	Failed	Failed	2222.2	1955.4
32770	63504	Failed	Failed	6508.5	5597.3

TABLE 6.5
Grid-Wide problem family – iterations

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
1026	2096	26	25	25	24
2050	4208	74	Failed	36	31
4098	8432	Failed	Failed	40	35
8194	16880	Failed	97	46	39
16386	33776	Failed	Failed	53	47
32770	67568	Failed	Failed	60	54

Grid-Wide, Mesh-1 and Netgen-Lo, were respectively, 39%, 22%, 18%, 29% and 38%. Time per iteration increased in all but three instances. The average increases in time per iteration for classes Grid-Density-16, Grid-Long, Grid-Wide, Mesh-1 and Netgen-Lo, were respectively, 3%, -3%, 5%, 10.6% and 4.7%.

- Though the fixed step size approach used in this study does not guarantee global convergence, it did, for these instances, work well. This can perhaps be explained by the fact that the iterates in these experiments were always "well centered." Tsuchiya & Monteiro [18] have shown that if the iterates remain in some sense well centered, then a step size asymptotically approaching 1 will produce a superlinear globally convergent affine scaling algorithm.

6.1. Default parameter setting. The performance of DLNET in the examples described earlier in this section does not fully reflect the code's capability. In the following, we run DLNET using the version 2.2a default parameter setting. With those settings the code uses the "fixed" step size rule, checks for optimality using the primal basic optimality checking [13] at every iteration, and the maximum flow optimality checking, using Indicator 3 with parameters $\kappa_0 = 0.55$ and $\kappa_1 = 0.95$, beginning at DAS iteration 25 with an initial frequency of 25 iterations. When $\mathcal{I}^k < 0.001$ the frequency is reduced to 5 iterations and then to every iteration when $\mathcal{I}^k < 0.0000001$. We require a primal-dual optimal integer solution to stop with either optimality test.

Instances identical to those used in the indicator/step-size pair runs were generated for these runs. For each problem family, a single table summarizes the runs, listing iterations and running times, and the CPU time ratios between version 1.4b (used in [13]) and version 2.2a of DLNET. Besides the new indicator, version 2.2a was made more efficient by rewriting several sections of the code. Table 6.11 summarizes the runs for problem family Grid-Density-16. Table 6.12 summarizes the runs for problem family Grid-Long. Table 6.13 summarizes the runs for problem family Grid-Wide. Table 6.14 summarizes the runs for problem family Mesh-1. Table 6.15 summarizes the runs for problem family Netgen-Lo. We make the following remarks about the computational experiment:

- As we intended to show, the default parameters for DLNET produce improved running times without compromising robustness.

TABLE 6.6
Grid-Wide problem family – CPU seconds

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
1026	2096	9.4	8.6	8.4	8.0
2050	4208	104.5	Failed	31.7	26.2
4098	8432	Failed	Failed	93.6	79.1
8194	16880	Failed	1566.6	301.6	237.4
16386	33776	Failed	Failed	815.7	709.9
32770	67568	Failed	Failed	2095.9	1735.8

TABLE 6.7
Mesh-1 problem family – iterations

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
256	512	Failed	Failed	16	16
1024	2048	Failed	Failed	22	20
4096	8192	Failed	Failed	40	31
16384	32768	Failed	Failed	62	52
65536	131072	Failed	Failed	60	47

- As compared to version 1.4b of the code, the latest version was faster on all but two instances, where version 1.4b was faster because it took fewer iterations. For the largest instances in each class, the decrease in running time went from a low of 8% to a high of 47%.

7. Concluding remarks. In this paper, we proposed three indicators based on new convergence results for the affine scaling algorithm for linear programming. We described the implementation of one such indicator, used to aid the identification of the optimal primal-dual structure of a minimum cost network flow problem. These indicators can be adapted to general linear programming. The experimental study suggests that the new indicator is more robust than an indicator previously used in the network code DLNET. The experiments further show that, for the instances tested, the effectiveness of the indicator is not as sensitive to step size as suggested by theory.

TABLE 6.8
Mesh-1 problem family – CPU seconds

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
256	512	Failed	Failed	1.1	1.0
1024	2048	Failed	Failed	9.1	7.7
4096	8192	Failed	Failed	197.6	136.3
16384	32768	Failed	Failed	2651.8	2017.1
65536	131072	Failed	Failed	16332.5	11688.4

TABLE 6.9
Netgen-Lo problem family – iterations

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
1024	8214	32	28	33	27
2048	16414	45	34	45	35
4096	32858	Failed	Failed	47	36
8192	65734	Failed	Failed	55	43
16384	131409	64	48	62	47
32768	262903	Failed	Failed	79	54

TABLE 6.10
Netgen-Lo problem family – CPU seconds

		Dual Slacks		Indicator 3	
Vertices	Edges	Dynamic	Fixed	Dynamic	Fixed
1024	8214	34.3	30.1	35.0	27.8
2048	16414	118.8	89.5	117.1	86.9
4096	32858	Failed	Failed	328.7	248.5
8192	65734	Failed	Failed	1104.0	821.6
16384	131409	3993.2	2819.4	4410.9	3202.3
32768	262903	Failed	Failed	13760.0	8580.8

TABLE 6.11
Grid-Density-16 problem family – default settings

Vertices	Edges	Iterations	v2.2a CPU seconds	v1.4b / v2.2a CPU ratio
1024	16384	48	110.2	1.30
2048	32768	51	304.7	1.43
4096	65536	55	808.1	1.26
8192	131072	71	2536.1	1.21
16384	262144	65	5847.9	1.17
32768	524288	66	14077.2	1.89

TABLE 6.12
Grid-Long problem family – default settings

Vertices	Edges	Iterations	v2.2a CPU secs.	v1.4b / v2.2a CPU ratio
1026	2000	29	9.7	1.59
2050	3984	37	33.8	1.49
4098	7952	50	138.9	1.43
8194	15888	59	409.4	1.43
16386	31760	78	1578.6	1.50
32770	63504	101	4525.8	1.44

TABLE 6.13
Grid-Wide problem family – default settings

Vertices	Edges	Iterations	v2.2a CPU seconds	v1.4b / v2.2a CPU ratio
1026	2096	25	7.5	1.37
2050	4208	31	22.0	1.52
4098	8432	35	68.4	1.28
8194	16880	38	166.2	1.28
16386	33776	47	521.0	1.47
32770	67568	54	1110.6	1.36

TABLE 6.14
Mesh-1 problem family – default settings

Vertices	Edges	Iterations	v2.2a CPU seconds	v1.4b / v2.2a CPU ratio
256	512	15	1.0	1.49
1024	2048	20	6.4	1.29
4096	8192	30	117.7	0.93
16384	32768	52	1757.7	0.68
65536	131072	47	8594.6	1.37

TABLE 6.15
Netgen-Lo problem family – default settings

Vertices	Edges	Iterations	v2.2a CPU seconds	v1.4b / v2.2a CPU ratio
1024	8214	27	28.4	1.34
2048	16414	35	83.0	1.29
4096	32858	37	235.3	1.35
8192	65734	44	693.7	1.28
16384	131409	49	2260.7	1.31
32768	262903	55	6024.9	1.09

Acknowledgment. The work of one of the authors (T. Tsuchiya) was based on research supported by the Overseas Research Scholars of the Ministry of Education, Science and Culture of Japan.

REFERENCES

[1] I. Adler, N. Karmarkar, M. Resende, and G. Veiga, *Data structures and programming techniques for the implementation of Karmarkar's algorithm*, ORSA Journal on Computing, 1 (1989), pp. 84–106.

[2] ———, *An implementation of Karmarkar's algorithm for linear programming*, Mathematical Programming, 44 (1989), pp. 297–335.

[3] I. Dikin, *Iterative solution of problems of linear and quadratic programming*, Soviet Mathematics Doklady, 8 (1967), pp. 674–675.

[4] ———, *Determination of interior point of a system of linear inequalities*, tech. report, Siberian Energy Institute, Irkutsk, USSR, 1991.

[5] DIMACS, *The first DIMACS international algorithm implementation challenge: The benchmark experiments*, tech. report, DIMACS, New Brunswick, NJ, 1991.

[6] L. Hall and R. Vanderbei, *Two-thirds is sharp for affine scaling*, Tech. Report SOR-92/9, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ 08544 USA, 1992.

[7] N. Karmarkar and K. Ramakrishnan, *Computational results of an interior point algorithm for large scale linear programming*, Mathematical Programming, 52 (1991), pp. 555–586.

[8] C. Monma and A. Morton, *Computational experiments with a dual affine variant of Karmarkar's method for linear programming*, Operations Research Letters, 6 (1987), pp. 261–267.

[9] R. Monteiro, I. Adler, and M. Resende, *A polynomial-time primal-dual affine scaling algorithm for linear and convex quadratic programming and its power series extension*, Mathematics of Operations Research, 15 (1990), pp. 191–214.

[10] R. Monteiro, T. Tsuchiya, and Y. Wang, *A simplified global convergence proof of the affine scaling algorithm*, tech. report, Department of Systems and Industrial Engineering, University of Arizona, Tucson, AZ 85721, 1992.

[11] M. Resende and G. Veiga, *Computational study of two implementations of the dual affine scaling algorithm*, tech. report, AT&T Bell Laboratories, Murray Hill, NJ, 1990.

[12] ———, *An implementation of the dual affine scaling algorithm for minimum cost flow on bipartite uncapaciated networks*, tech. report, AT&T Bell Laboratories, Murray Hill, NJ, 1990. To appear in SIAM Journal on Optimization.

[13] ———, *An efficient implementation of a network interior point method*, tech. report, AT&T Bell Laboratories, Murray Hill, NJ, 1992. To appear in DIMACS Series in Discrete Mathematics and Theoretical Computer Science.

[14] R. Saigal, *A three step quadratically convergent implementation of the primal affine scaling method*, tech. report, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109-2217 USA, 1993.

[15] M. Todd and B. Burrell, *An extension to Karmarkar's algorithm for linear programming using dual variables*, Algorithmica, 1 (1986), pp. 409–424.

[16] T. Tsuchiya, *Global convergence of the affine scaling methods for degenerate linear programming problems*, Mathematical Programming, 52 (1991), pp. 377–404.

[17] ———, *Global convergence property of the affine scaling methods for primal degenerate linear programming problems*, Mathematics of Operations Research, 17 (1992), pp. 527–557.

[18] T. Tsuchiya and R. Monteiro, *Superlinear convergence of the affine scaling algorithm*, tech. report, Institute of Statistical Mathematics, Tokyo, 106 Japan, 1992.

[19] T. Tsuchiya and M. Muramatsu, *Global convergence of the long-step affine scaling algorithm for degenerate linear programming problems*, tech. report, The Institute of Statistical Mathematics, Tokyo, January 1992.