Convergent Lagrangian heuristics for nonlinear minimum cost network flows

Torbjörn Larsson,* Johan Marklund,† Caroline Olsson,‡ Michael Patriksson§

June 5, 2007

Abstract

We consider the separable nonlinear and strictly convex single-commodity network flow problem (SSCNFP). We develop a computational scheme for generating a primal feasible solution from *any* Lagrangian dual vector; this is referred to as "early primal recovery". It is motivated by the desire to obtain a primal feasible vector before convergence of a Lagrangian scheme; such a vector is not available from a Lagrangian dual vector unless it is optimal. The scheme is constructed such that if we apply it from a sequence of Lagrangian dual vectors that converge to an optimal one, then the resulting primal (feasible) vectors converge to the unique optimal primal flow vector. It is therefore also a convergent Lagrangian heuristic, akin to those primarily devised within the field of combinatorial optimization but with the contrasting and striking advantage that it is guaranteed to yield a primal optimal solution in the limit. Thereby we also gain access to a new stopping criterion for any Lagrangian dual algorithm for the problem, which is of interest in particular if the SSCNFP arises as a subproblem in a more complex model. We construct instances of convergent Lagrangian heuristics that are based on graph searches within the residual graph, and therefore are efficiently implementable; in particular we consider two shortest path based heuristics that are based on the optimality conditions of the original problem. Numerical experiments report on the relative efficiency and accuracy of the various schemes.

1 Introduction

When solving large-scale, structured, convex nonlinear optimization problems through decomposition and coordination it is very often the case that the generation of an optimal solution to the subproblem is a complex task to accomplish; at the same time it is both feasible theoretically and advantageous computationally to provide only a rough feasible solution (rather than an optimal solution) to the subproblem, since it is more important to provide information that aid in improving the incumbent solution through the solution to the corresponding "master" problem. Devising effective termination criteria for the subproblem, together with the generation of a rough feasible solution, is therefore of interest for some classes of problems.

In this article we focus on a classic subproblem, namely the separable nonlinear and strictly convex single-commodity network flow problem (SSCNFP). It arises naturally as a subproblem when solving a problem with multiple commodities, a non-separable objective function, and/or when the original problem has additional (side) constraints. It arises also as a subproblem in several types of algorithms for network flow optimization in transportation planning, such as in Evans' algorithm and Gauss–Seidel (cyclic decomposition) type methods for traffic equilibrium problems, cf. [Pat94].

A standard means to solve the SSCNFP is by extracting a primal optimal solution from an optimal solution to its Lagrangian dual; cf., for example, [BHT87, Ven91, CeZ97]. The Lagrangian dual problem is unconstrained and also differentiable in many cases, and therefore efficiently solvable by a large number of unconstrained convex optimization algorithms that exploit the sparsity of the problem. Unfortunately

*Department of Mathematics, Linköping University, SE-581 83 Linköping, Sweden. E-mail: tolar@mai.liu.se
†Department of Industrial Management and Logistics, Lund University, SE-221 00 Lund, Sweden. Email: Johan.Marklund@iml.lth.se
‡Department of Radio Physics, Göteborg University, SE-413 45 Gothenburg, Sweden. E-mail: caroline.olsson@vgregion.se
§Department of Mathematical Sciences, Chalmers University of Technology and Department of Mathematical Sciences, Göteborg University, SE-412 96 Gothenburg, Sweden. E-mail: mipat@math.chalmers.se

the Lagrangian dual problem can only supply a feasible network flow if we first solve it to optimality. This consequence is inherent in the dual problem's optimality conditions, which we recall is that the dual gradient is zero, while the dual gradient is, precisely, the network flow residual.

This fact motivates us to develop a simple computational scheme for generating a primal feasible solution from *any* Lagrangian dual vector, rather than just an optimal one. This is what we refer to as "early primal recovery". Such a scheme should also be constructed such that if we were to apply it from a sequence of Lagrangian dual vectors that converge to an optimal one, then the resulting primal (feasible) vectors should converge to an optimal primal flow vector. This article provides a class of such schemes, based on graph searches. We compare them on some strictly convex quadratic network flow problems generated by principles given by Ohuchi and Kaji [OhK84] (bipartite graphs) and by NETGEN ([KNS74]) (general graphs).

Previously, only two Lagrangian heuristics for the problem SSCNFP have been presented, namely those of Ventura [Ven91] and Curet [Cur93]. Both are based on the solution of linear SSCNFPs, and they are not convergent in the sense that if they are applied from a sequence of Lagrangian dual vectors that converge to an optimal one, then the resulting primal vectors generally will not converge to an optimal primal flow vector; we, however, construct a convergent heuristic based on a combination of the two.

Plentiful of further applications of the large-scale SSCNFP exist where the typical solution algorithm is of a Lagrangian dual nature, and where the efficient construction of a primal feasible and near-optimal solution would be most useful; see, for example, the text book by Censor and Zenios [CeZ97], which includes several computational applications in finance, medicine (image restoration tomography and radiation therapy), industrial planning, and economics.

The use of the "early primal recovery" concept has so far received little attention in the subject of convex optimization, where however the strong duality property makes it quite appealing. Instead, its popularity has been much stronger in the fields of combinatorial, integer and discrete optimization, where it is typically referred to by the term "Lagrangian heuristics" (see Fisher [Fis81]). These problem areas include several with constraint structures similar to those of the present problem, however with additional integrality requirements on the link flow variables. They are however ones where the success of primal feasibility schemes is limited by the fact that the strong duality property seldom holds. Our contribution should therefore be seen also as a belated one in an area where its likely success is much greater. For a recent analysis of Lagrangian heuristics in the field of integer programming, see [LaP06].

In the next section we develop the model SSCNFP and its Lagrangian dual. We briefly present the principle of generating the optimal flow from the solution to the Lagrangian dual problem through the analysis of the problem's primal–dual global optimality conditions, and discuss algorithm principles.

In Section 3 we then present a class of procedures for generating a primal feasible flow from a primal infeasible but Lagrangian optimal flow. They are either based on performing searches within the residual graph or on the solution of a linear optimization problem over the set of feasibility-inducing flow adjustments. We show that if a sequence of Lagrangian optimal flows is generated such that it converges to the optimal solution to the SSCNFP then the sequence of vectors produced by a primal feasibility scheme is also optimizing, provided that the scheme "almost" acts as an Euclidean projection onto the network flow polytope. This fact is used to prove the correctness of our schemes.

In Section 4 we report on experiments with our primal recovery schemes on a number of test examples with different numbers of nodes, numbers of sources and sinks, and different levels of sparsity, feasibility, and cost nonlinearity. Among our findings a main conclusion is that it is possible to efficiently generate primal feasible flows from a dual vector.

2 Preliminaries

2.1 The optimization model

Let $\boldsymbol{E} \in \mathbb{R}^{m\times n}$ be the node–link incidence matrix of a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ with m nodes and n links. Let $\boldsymbol{d} \in \mathbb{R}^m$ denote the vector of demands, satisfying $\mathbf{1}^{\mathrm{T}}\boldsymbol{d} = 0$. Our sign convention is that a source (respectively, sink) node i has d_i negative (respectively, positive), and, therefore, element (i, j) of $\boldsymbol{E}$ is -1 if link j has node i as its source node, 1 if node i is its sink node, or otherwise zero. The conservation of flow at node i is described by the following linear equations:

$$\sum_{k:(k,i)\in\mathcal{L}} v_{ki} - \sum_{k:(i,k)\in\mathcal{L}} v_{ik} = d_i, \qquad i = 1, \dots, m,$$

or, in other words, $\boldsymbol{E}\boldsymbol{v} = \boldsymbol{d}$.

Identifying $l = (i,k)$ for $(i,k) \in \mathcal{L}$, we state the separable single-commodity network flow problem as follows: find $\boldsymbol{v} \in \mathbb{R}^n$ to

$$\underset{\boldsymbol{v}}{\text{minimize}}\ \phi(\boldsymbol{v}) := \sum_{l\in\mathcal{L}} \phi_l(v_l), \tag{1a}$$

$$\text{subject to } \boldsymbol{E}\boldsymbol{v} = \boldsymbol{d}, \tag{1b}$$

$$v_l \in X_l, \qquad l \in \mathcal{L}, \tag{1c}$$

where, for each $l \in \mathcal{L}$, $X_l \subset \mathbb{R}$ is a non-empty and closed interval, and the cost functions $\phi_l : \mathbb{R} \to \mathbb{R}$ for all links $l \in \mathcal{L}$ satisfy the condition that

$$\text{the function } \phi_l \text{ is in } C^1 \text{ and strictly convex on } \mathbb{R}. \tag{2}$$

Under these conditions, and providing that a feasible solution exists, the problem (1) has a unique link flow solution, by Weierstrass' Theorem and the strict convexity of the objective function. We let F denote the set of feasible solutions to the problem (1), and $X := \prod_{l\in\mathcal{L}} X_l$ denote the box given by the bounds on the link flows.

2.2 The Lagrangian dual problem and optimality conditions

We introduce the Lagrange multiplier vector $\boldsymbol{\pi} \in \mathbb{R}^m$, and define the Lagrangian dual function

$$\begin{aligned} \eta(\boldsymbol{\pi}) &:= \underset{\boldsymbol{v}}{\text{minimum}}\ \left\{\phi(\boldsymbol{v}) - \boldsymbol{\pi}^{\mathrm{T}}(\boldsymbol{E}\boldsymbol{v} - \boldsymbol{d})\right\}, \\ &\qquad \text{subject to } v_{ik} \in X_{ik}, \qquad (i,k) \in \mathcal{L} \\ &= \boldsymbol{d}^{\mathrm{T}}\boldsymbol{\pi} + \sum_{(i,k)\in\mathcal{L}} \underset{v_{ik}\in X_{ik}}{\text{minimum}}\ \{\phi_{ik}(v_{ik}) - [\pi_k - \pi_i]v_{ik}\}. \end{aligned} \tag{3}$$

Before moving on, we state a result on the properties of the dual of a convex function over an interval; this result will then be used to describe the properties of the function η. The result is a special case of that for general conjugate convex functions, as covered in [Roc70], for example.

Proposition 1 (the Fenchel dual of a convex one-variable problem) *Let $\phi : \mathbb{R} \to \mathbb{R}$ be a strictly convex function, $X \subseteq \mathbb{R}$ be a non-empty and closed interval, and let*

$$q(s) = \underset{x\in X}{\text{infimum}}\ \{\phi(x) - sx\}, \qquad s \in \mathbb{R}. \tag{4}$$

(a) *We have that*

$$\underset{s}{\text{supremum}}\ \{q(s) + sx\} = \begin{cases} \phi(x), & x \in X, \\ \infty, & \textit{otherwise}, \end{cases} \tag{5}$$

and the following statements are equivalent for any pair (x,s) of scalars:

(1) *$sx = \phi(x) - q(s)$;*

(2) *x attains the infimum in (4);*

(3) *s attains the supremum in (5).*

(b) *Assume that for each $s \in \mathbb{R}$, the infimum in (4) is uniquely attained by, say, $x(s)$. Then, q is real-valued and differentiable, and we have that*

$$q'(s) = -x(s), \qquad s \in \mathbb{R}. \tag{6}$$

Proof. See [BeT89, Lemma 5.5.1] or [Ber98, Proposition 9.6(b)]. The appearance of the derivative follows also from Danskin's Theorem. □

The following result states that solving the Lagrangian dual problem, to

$$\underset{\boldsymbol{\pi}}{\text{maximize}}\ \eta(\boldsymbol{\pi}) := \boldsymbol{d}^{\mathrm{T}}\boldsymbol{\pi} + \sum_{(i,k)\in\mathcal{L}} q_{ik}(\pi_k - \pi_i), \tag{7a}$$

$$\text{subject to } \boldsymbol{\pi} \in \mathbb{R}^m, \tag{7b}$$

where

$$q_{ik}(s) := \underset{v_{ik}\in X_{ik}}{\text{infimum}} \{\phi_{ik}(v_{ik}) - sv_{ik}\}, \qquad s \in \mathbb{R},$$

yields a dual solution complementary to the unique primal optimal solution. The result is quite special for separable problems, also known as *monotropic programs* ([Roc84]).

THEOREM 2 (strong duality) *Suppose that there exists a feasible solution to* (1), *and that the conditions in* (2) *hold.*

(a) *There exists a pair* $(\boldsymbol{v}^*, \boldsymbol{\pi}^*)$ *which is optimal in* (1) *and* (7), *respectively, and strong duality holds, that is,* $\phi(\boldsymbol{v}^*) = \eta(\boldsymbol{\pi}^*)$.

(b) *Let for each* $(i,k) \in \mathcal{L}$

$$\phi^+_{ik}(v_{ik}) = \begin{cases} \phi'_{ik}(v_{ik}), & v_{ik} < u_{ik}, \\ \infty, & \textit{otherwise}, \end{cases}$$

and

$$\phi^-_{ik}(v_{ik}) = \begin{cases} \phi'_{ik}(v_{ik}), & v_{ik} > 0, \\ -\infty, & \textit{otherwise}. \end{cases}$$

A primal–dual pair $(\boldsymbol{v}^*, \boldsymbol{\pi}^*)$ *which is feasible in the respective problem is also optimal if and only if the complementarity slackness condition*

$$\phi^-_{ik}(v^*_{ik}) \leq \pi^*_k - \pi^*_i \leq \phi^+_{ik}(v^*_{ik}), \qquad (i,k) \in \mathcal{L}, \tag{8}$$

holds.

PROOF. (a) The problem (1) is a special case of a monotropic programming problem (see [Roc84] for its definition). The corresponding strong duality result for monotropic programs can be found in Minty [Min60], Rockafellar [Roc70, Roc84], or Bertsekas [Ber98, Proposition 9.4]. These cited results do not, however, assert that the primal or dual problems have optimal solutions. However, our stronger assumptions (2) imply the existence of optimal solutions to both problems (cf. Weierstrass' Theorem for the primal problem, and [Ber98, Proposition 9.3] for the dual problem).

(b) The condition (8) is equivalent to the complementarity conditions which is also equivalent to the property that $\phi(\boldsymbol{v}^*) = \eta(\boldsymbol{\pi}^*)$ holds. This equality is of course equivalent to the optimality of each vector in its respective problem. □

We also have the following classic optimality condition. In the result below, we note that a cycle is not blocked at $\boldsymbol{v}^*$ if and only if we can send some positive and negative amount of flow along it, without falling outside of the set X. We let $C = C^+ \cup C^-$, where the two disjoint subsets C^+ and C^- are defined by those links in the definition of a cycle C which are forward and backward links, respectively.

PROPOSITION 3 (optimality conditions) *The vector* $\boldsymbol{v}^*$ *is an optimal solution to* (1) *if, and only if, for every simple cycle* C *in the network which is not blocked at* $\boldsymbol{v}^*$,

$$\sum_{(i,k)\in C^+} \phi^+_{ik}(v^*_{ik}) - \sum_{(i,k)\in C^-} \phi^-_{ik}(v^*_{ik}) \geq 0. \tag{9}$$

PROOF. The proof is simply obtained by making use of the Flow Decomposition Theorem (cf. [AMO93, Section 3.5]), that is, the Representation Theorem as applied to network flow polyhedra. Based on it, we can claim that any feasible direction from $\boldsymbol{v}^*$ is a nonnegative linear combination of simple cycles, and representing a feasible direction in this way, we obtain the appearance (9) of the above variational inequality. This completes the proof. (A detailed proof is found in [Ber98, Proposition 9.1]. In fact, that result allows for the functions ϕ_{ik} to be non-differentiable.) □

The result states that the sum of the "reduced costs," that is, the sum of cost derivatives, are nonnegative along any cycle; this is a general property of optimal solutions in minimum cost network flow problems (see, for example, [AMO93, Section 9.3]), and is a special case of the variational inequality characterization of an optimal solution to a convex problem.

We now turn to an important property of the dual problem for the purpose of computations: the gradient $\nabla\eta(\boldsymbol{\pi})$ has a special form which makes dual searches simple. Thanks to the properties (2) of the functions ϕ_{ik}, we satisfy the conditions of Proposition 1(b) and therefore η is differentiable. Moreover,

$$\begin{aligned}
\frac{\partial\eta(\boldsymbol{\pi})}{\partial\pi_i} &= \sum_{(m,n)\in\mathcal{L}} \frac{\partial q_{mn}(\pi_n-\pi_m)}{\partial\pi_i} + d_i \\
&= \sum_{k:(k,i)\in\mathcal{L}} q'_{ki}(\pi_i-\pi_k) - \sum_{k:(i,k)\in\mathcal{L}} q'_{ik}(\pi_k-\pi_i) + d_i \\
&= -\sum_{k:(k,i)\in\mathcal{L}} v_{ki}(\pi_i-\pi_k) + \sum_{k:(i,k)\in\mathcal{L}} v_{ik}(\pi_k-\pi_i) + d_i \\
&= -(\boldsymbol{E}\boldsymbol{v}(\boldsymbol{\pi})-\boldsymbol{d})_i, \qquad i\in\mathcal{N}. \qquad (10)
\end{aligned}$$

This result comes as no surprise: the partial derivative of η with respect to π_i equals the violation of the i^{th} flow conservation constraint in (1b); in the context of network flows, the violation of flow conservation is also called the *divergence* (or, *surplus* or *imbalance*) of node i.

The necessary and sufficient optimality conditions for the dual problem, being convex, differentiable and unconstrained, is that

$$\nabla\eta(\boldsymbol{\pi}^*) = \mathbf{0}^m. \qquad (11)$$

In other words, dual optimality means feasibility in the Lagrangian relaxed primal constraints. Further, the unique optimal primal solution equals $\boldsymbol{v}(\boldsymbol{\pi}^*) := (v_{ik}(\pi_k^*-\pi_i^*))_{(i,k)\in\mathcal{L}}$.

It follows also from the optimality conditions of the primal–dual problem that the link flow vector $\boldsymbol{v}(\boldsymbol{\pi})$ of these optimal solutions to the n problems of the form (3) is an infeasible flow whenever the dual price vector $\boldsymbol{\pi}$ is non-optimal. The next section is devoted to the design of procedures for generating feasible flows from a (non-optimal) vector $\boldsymbol{v}(\boldsymbol{\pi})$, and which may be used when the dual search is terminated finitely.

2.3 Algorithms for the Lagrangian dual problem

In order to reach a dual optimal solution, it seems natural to utilize the primal–dual relations established above, and therefore to try to balance node i's flow through the manipulation of the dual price π_i. Performing a restricted dual search in which η is maximized over the single dual price π_i, that is, performing a line search in the i^{th} dual coordinate, is equivalent to satisfying the i^{th} equality constraint (in a Lagrangian optimal manner). As the dual function is concave, $\frac{\partial\eta(\boldsymbol{\pi})}{\partial\pi_i}$ is a non-increasing function of π_i. Suppose that $\frac{\partial\eta(\boldsymbol{\pi})}{\partial\pi_i} > 0$; this is the same as saying that

$$g_i(\boldsymbol{\pi}) := -\frac{\partial\eta(\boldsymbol{\pi})}{\partial\pi_i} = \sum_{k:(k,i)\in\mathcal{L}} v_{ki}(\pi_i-\pi_k) - \sum_{k:(i,k)\in\mathcal{L}} v_{ik}(\pi_k-\pi_i) - d_i < 0,$$

or, in other words, that too much net flow leaves (or, too little net flow reaches) the node. In order to balance the flow, a line search in the value of π_i would then increase its value.[1] There is always a solution to the line search problem, because a dual line search is a restriction (to a particular one-dimensional subspace) of the dual problem (7), which we know has a solution by Theorem 2(a). (An independent motivation can be found in [BeT89, p. 398] or [Ber98, p. 431].)

In order to complete an algorithmic description, we must decide on two things: which node prices to update at any iteration; and the accuracy by which line searches are performed. Updating one node price at a time leads to a coordinate-wise search procedure, and thus has a lot in common with Gauss–Seidel and Jacobi algorithms for the solution of a system of nonlinear equations arising from the optimality conditions of a differentiable, unconstrained optimization problem, but it is also different, for the following reasons. Although the dual function η is concave and differentiable, it is in general *not* strictly concave; this is a convergence requirement for the Gauss–Seidel algorithm. Further, we cannot a priori guarantee that the sequence of iterates is bounded, because the level set of the dual function η is

[1] We can also see this as follows. The function $\pi_i \mapsto q_{ki}(\pi_i-\pi_k)$ is concave; therefore, its derivative function $\pi_i \mapsto q'_{ki}(\pi_i-\pi_k)$ is non-increasing. That also means that the function $\pi_i \mapsto v_{ki}(\pi_i-\pi_k)$, by identity, is non-decreasing. Therefore, raising the value of π_i will increase the flow into node i. Similarly, the function $\pi_i \mapsto v_{ik}(\pi_k-\pi_i)$ is a non-increasing function, so that raising the value of π_i means simultaneously decreasing the flow out of node i.

always unbounded: if a common constant is added to all multiplier values in $\boldsymbol{\pi}$, then the dual function value is unchanged. This potential problem can however be eliminated by fixing the multiplier value for one node to any constant. This is possible due to the linear dependence of the rows in the system $(\boldsymbol{E}, \boldsymbol{d})$: summing the rows yields all zeroes. In effect, what we are doing by setting one dual price to zero (or any other value) is that we eliminate a redundant primal constraint.

Since, from Theorem 2(a), we know that the unconstrained Lagrangian dual problem (7) has an optimal solution, we can invoke a result from Iusem [Ius03] which for our problem would state that the steepest ascent method with (inexact) line searches converges to an optimal solution to (7). Based on this basic result, we can develop quasi-Newton methods with ensured convergence of the sequence $\{\boldsymbol{\pi}^\tau\}$.

Conjugate direction methods are popular for the problem (7), especially for the case of quadratic objective functions ϕ. Lin and Pang [LiP87] establish the convergence of a dual ascent algorithm for a class of strictly convex quadratic problems that includes the quadratic version of our problem (1). Their analysis is performed without any modification of the dual problem by fixing a dual variable to zero, so dual convergence cannot be ensured for their problem; convergence of the primal sequence $\{\boldsymbol{v}(\boldsymbol{\pi})\}$ is however ensured. We assume however that a node price has been fixed, and consider a particular instance of the conjugate direction method as a special case of their methodology: At some iteration τ suppose $\boldsymbol{\pi}^\tau$ is given, and let the search direction be given by

$$\boldsymbol{p}^\tau := \begin{cases} \nabla\eta(\boldsymbol{\pi}^\tau) + \beta^\tau \boldsymbol{p}^{\tau-1}, & \text{if } \tau \,(\operatorname{mod} m) \neq 0, \\ \nabla\eta(\boldsymbol{\pi}^\tau), & \text{otherwise}, \end{cases} \tag{12a}$$

where

$$\beta^\tau := \frac{\nabla\eta(\boldsymbol{\pi}^\tau)^{\mathrm{T}}[\nabla\eta(\boldsymbol{\pi}^\tau) - \nabla\eta(\boldsymbol{\pi}^{\tau-1})]}{\nabla\eta(\boldsymbol{\pi}^{\tau-1})^{\mathrm{T}}\nabla\eta(\boldsymbol{\pi}^{\tau-1})}. \tag{12b}$$

This search direction formula, known as the Polak–Ribiere–Polyak formula, has been reported to yield a better performance than some other possible conjugate gradient formulas for the problem at hand (cf., e.g., [LiP87, Ven91]).

To define the next iterate, $\boldsymbol{\pi}^{\tau+1}$ a line search in the function η is performed in the direction of $\boldsymbol{p}^\tau$. This line search problem can be shown to be equivalent to Lagrangian optimally fulfilling an aggregated form of the flow conservation constraints: in fact, the optimal step length, ℓ^τ, is the Lagrange multiplier in the problem to

$$\underset{\boldsymbol{v}}{\text{minimize}}\ \phi(\boldsymbol{v}) - (\boldsymbol{\pi}^\tau)^{\mathrm{T}}(\boldsymbol{E}\boldsymbol{v} - \boldsymbol{d}), \tag{13a}$$

$$\text{subject to } \boldsymbol{p}^{\mathrm{T}}(\boldsymbol{E}\boldsymbol{v} - \boldsymbol{d}) = 0, \tag{13b}$$

$$v_l \in X_l, \qquad l \in \mathcal{L}. \tag{13c}$$

This problem is a special strictly convex problem with only one linear constraint and variable bounds, and is often referred to as a simple resource allocation problem. It is solved efficiently through its Lagrangian relaxation and determination of its only multiplier, namely the optimal step in the line search. In the special case when ϕ is quadratic, the Lagrangian dual function has a piecewise linear derivative; searching its break-points with, say, a median search method, a zero can be found in a time proportional to m. Some references for the solution to this line search problem are [HKL80, Bru84, IbK88]; a recent survey is also found in [Pat07].

We mention in brief that although we are dealing with an unconstrained dual problem, we can use the primal subproblem in a stopping criterion: since the value of $\|\boldsymbol{E}\boldsymbol{v}(\boldsymbol{\pi}^\tau) - \boldsymbol{d}\|/\max\{\|\boldsymbol{d}\|, 1\}$ tends to zero when $\tau \to \infty$, a possible stopping criterion therefore is to terminate the search if

$$\|\boldsymbol{E}\boldsymbol{v}(\boldsymbol{\pi}^\tau) - \boldsymbol{d}\| \leq \varepsilon \max\{\|\boldsymbol{d}\|, 1\}, \tag{14}$$

where $\varepsilon > 0$ is small.

3 Early primal recovery

3.1 Introduction

Suppose that the (non-optimal) dual vector $\widehat{\boldsymbol{\pi}} \in \mathbb{R}^m$ is given. According to the optimality conditions (11), the primal optimal solution is given by $\boldsymbol{v}(\boldsymbol{\pi}^*)$, that is, the Lagrangian optimal solution, for any dual optimal solution $\boldsymbol{\pi}^*$, and it is moreover characterized by the fact that $\boldsymbol{v}(\boldsymbol{\pi}^*)$ is a primal feasible solution. Hence, for a non-optimal solution such as $\widehat{\boldsymbol{\pi}}$, the corresponding Lagrangian optimal vector $\boldsymbol{v}(\widehat{\boldsymbol{\pi}})$ is always an *infeasible* flow. The purpose of a *primal feasibility scheme* is to generate a feasible flow, perhaps by adjusting the vector $\boldsymbol{v}(\widehat{\boldsymbol{\pi}})$; such a scheme is, preferably, computationally cheap, if we wish to apply it regularly, and more importantly the scheme should have the property that as the dual vector approaches an optimal one, so should the feasible flow generated by the scheme. (If we utilize $\boldsymbol{v}(\widehat{\boldsymbol{\pi}})$ then it means in particular that it should be a *local*—we may also call it *conservative*—scheme, in the sense that the move from $\boldsymbol{v}(\widehat{\boldsymbol{\pi}})$ is minimal while ensuring feasibility; cf. [LaP06].) We will formalize this requirement below. (The notation $\operatorname{Proj}_F(\boldsymbol{z})$ means the unique vector of F that minimizes the Euclidean distance to the vector $\boldsymbol{z}$.)

Definition 4 (a projection-like mapping) *Let $P_F : X \to F$ be a vector-valued mapping which assigns any vector in X to a unique vector in F, the feasible set of* (1). *Consider a sequence $\{\boldsymbol{z}^\tau\} \subset X$. If the mapping P_F has the property that*

$$\operatorname{Proj}_F(\boldsymbol{z}^\tau) - \boldsymbol{z}^\tau \to \mathbf{0}^n \qquad \Longrightarrow \qquad P_F(\boldsymbol{z}^\tau) - \boldsymbol{z}^\tau \to \mathbf{0}^n, \tag{15}$$

we then say that the mapping is projection-like *onto F.* □

The interest in such a projection-like mapping is the following. If we consider a sequence $\{\boldsymbol{\pi}^\tau\}$ generated by a dual algorithm, then the corresponding sequence of primal feasible solutions constructed by a projection-like feasibility scheme applied to the subproblem solutions $\boldsymbol{v}(\boldsymbol{\pi}^\tau)$ will, in the limit, produce an optimal solution to the primal problem. (Notice that the definition poses no requirement on the convergence of the dual sequence itself.) This is shown next.

Proposition 5 (primal optimal solutions from projection-like feasibility schemes) *Consider a sequence $\{\boldsymbol{\pi}^\tau\}$ with the property that $\boldsymbol{v}(\boldsymbol{\pi}^\tau) \to \boldsymbol{v}^*$, that is, one which induces primal convergence of the Lagrangian subproblem solutions. Consider the sequence $\{\boldsymbol{v}^\tau\} = \{P_F(\boldsymbol{v}(\boldsymbol{\pi}^\tau))\}$ of primal feasible solutions generated by a projection-like feasibility scheme initiated at $\boldsymbol{v}(\boldsymbol{\pi}^\tau)$. Then, $\boldsymbol{v}^\tau \to \boldsymbol{v}^*$.*

Proof. It holds that

$$\|\boldsymbol{v}^\tau - \boldsymbol{v}^*\| = \|P_F(\boldsymbol{v}(\boldsymbol{\pi}^\tau)) - \boldsymbol{v}^*\| \le \|P_F(\boldsymbol{v}(\boldsymbol{\pi}^\tau)) - \boldsymbol{v}(\boldsymbol{\pi}^\tau)\| + \|\boldsymbol{v}(\boldsymbol{\pi}^\tau) - \boldsymbol{v}^*\|.$$

Because $\boldsymbol{v}(\boldsymbol{\pi}^\tau) \to \boldsymbol{v}^*$ holds, it follows by the continuity of the projection operation that $\operatorname{Proj}_F(\boldsymbol{v}(\boldsymbol{\pi}^\tau)) - \boldsymbol{v}(\boldsymbol{\pi}^\tau) \to \mathbf{0}^n$. Applying (15) with $\boldsymbol{z}^\tau := \boldsymbol{v}(\boldsymbol{\pi}^\tau)$, it therefore follows that $P_F(\boldsymbol{v}(\boldsymbol{\pi}^\tau)) - \boldsymbol{v}(\boldsymbol{\pi}^\tau) \to \mathbf{0}^n$. Hence, the right-hand side of the above inequality tends to zero as τ tends to infinity, and we are done. □

The projection-like property was first utilized by Larsson and Liu [LaL97] when devising primal feasibility schemes for linear multicommodity network flow problems.

It is clear from the proof that primal vectors other than the Lagrangian optimal solutions $\boldsymbol{z}^\tau := \boldsymbol{v}(\boldsymbol{\pi}^\tau)$ can be used in the primal feasibility scheme and still yield primal convergence, provided however that the vector $\boldsymbol{z}^\tau$ is optimizing in the limit. As an example, consider primal feasibility schemes which utilize the primal sequences that are generated within ε-relaxation methods (see [Ber98]); they are not optimal with respect to the Lagrangian subproblem, but as ε tends to zero they are optimizing in the limit.

That a projection-like feasibility scheme would be successful in reaching near-optimal primal solutions is also clear from the speed in which the primal (infeasible) sequence $\{\boldsymbol{v}(\boldsymbol{\pi}^\tau)\}$ converges to $\boldsymbol{v}^*$. According to a convergence theorem in [LuT93], the primal–dual convergence speed is linear for certain dual gradient algorithms, among which are a group of relaxation methods. The basis for this result is an *error bound*, which measures the distance from any dual vector to the set, Π^*, of optimal dual solutions in terms of a "residual" [in the unconstrained case the residual is $\eta(\boldsymbol{\pi}^\tau)$]. We cite such a result in [LuT93] below.

Theorem 6 (an error bound for the primal–dual pair (1), (7)) *Suppose that the objective function of the single-commodity network flow problem* (1) *is strongly convex with a Lipschitz continuous gradient and that it satisfies* (2).

(a) [convergence of the primal solution] *For any scalar $\gamma \le \phi(\boldsymbol{v}^*)$, there exist positive scalars δ and μ such that, for any $\boldsymbol{\pi} \in \mathbb{R}^m$ with $\eta(\boldsymbol{\pi}) \ge \gamma$ and $\|\nabla\eta(\boldsymbol{\pi})\| \le \delta$,*

$$\|\boldsymbol{v}(\boldsymbol{\pi}) - \boldsymbol{v}^*\| + \underset{\boldsymbol{\pi}^* \in \Pi^*}{\operatorname{minimum}} \{\|\boldsymbol{\pi} - \boldsymbol{\pi}^*\|\} \le \mu \|\nabla\eta(\boldsymbol{\pi})\|. \tag{16}$$

(b) [convergence of the duality gap] *For any scalar* $\gamma \leq \phi(\boldsymbol{v}^*)$, *there exist positive scalars* δ *and* μ' *such that, for any* $\boldsymbol{\pi} \in \mathbb{R}^m$ *with* $\eta(\boldsymbol{\pi}) \geq \gamma$ *and* $\|\nabla\eta(\boldsymbol{\pi})\| \leq \delta$,

$$\phi(\boldsymbol{v}^*) - \eta(\boldsymbol{\pi}) \leq \mu' \|\nabla\eta(\boldsymbol{\pi})\|^2. \tag{17}$$

Proof. See [LuT93, Theorem 4.1 and Corollary 4.1]. □

We see that the speed at which we approach the primal optimal solution is related to the speed at which the dual gradient tends to zero; further, a sequence of primal feasible flows generated from a projection-like feasibility scheme would inherit the linear speed of convergence of the sequence of Lagrangian optimal solutions.

In the following we introduce and establish several realizations of the feasibility scheme.

The next subsection collects primal feasibility schemes that are based on the solution of linear network flow optimization problems. The only hitherto published primal feasibility schemes are of this type. The scheme of Ventura [Ven91] is based on a minimum cost network flow problem; the scheme of Curet [Cur93] is a pure feasibility problem. We then provide two new feasibility schemes; the first is a combination of Ventura's and Curet's schemes, while the second is an improvement over Ventura's scheme. Both these new schemes are established to be projection-like.

3.2 Network flow optimization based realizations

Given a fixed flow $\widehat{\boldsymbol{v}} \in \mathbb{R}^n$ the *residual graph* describes the set of flow adjustments $\boldsymbol{\delta} \in \mathbb{R}^n$ for which $\widehat{\boldsymbol{v}} + \boldsymbol{\delta} \in F$, that is, vectors $\boldsymbol{\delta} \in \mathbb{R}^n$ satisfying

$$\boldsymbol{E}(\widehat{\boldsymbol{v}} + \boldsymbol{\delta}) = \boldsymbol{d}, \tag{18a}$$

$$\hat{\boldsymbol{v}} + \boldsymbol{\delta} \in X, \tag{18b}$$

A good feasibility scheme is one which is inexpensive in finding such a vector, and one which moreover has the projection-like property; among the properties of such a scheme is that $\|\boldsymbol{\delta}\|$ should approach zero as $\widehat{\boldsymbol{v}}$ approaches F.

3.2.1 Ventura's scheme

The feasibility scheme of Ventura [Ven91] appears to be the first one published. It solves a linearization of the primal problem at $\widehat{\boldsymbol{v}} := \boldsymbol{v}(\widehat{\boldsymbol{\pi}})$ over the residual graph; that is, the linear cost is $\boldsymbol{\delta} \mapsto \nabla\phi(\widehat{\boldsymbol{v}})^{\mathrm{T}}\boldsymbol{\delta}$:

$$\underset{\boldsymbol{\delta}}{\text{minimize}}\ \nabla\phi(\boldsymbol{v}(\widehat{\boldsymbol{\pi}}))^{\mathrm{T}}\boldsymbol{\delta}, \tag{19a}$$

$$\text{subject to } \boldsymbol{E}(\boldsymbol{v}(\widehat{\boldsymbol{\pi}}) + \boldsymbol{\delta}) = \boldsymbol{d}, \tag{19b}$$

$$\boldsymbol{v}(\widehat{\boldsymbol{\pi}}) + \boldsymbol{\delta} \in X. \tag{19c}$$

This linear program (which is equivalent to a linear minimum cost network flow problem) does *not* induce a projection-like scheme, and therefore is not guaranteed to provide near-optimal primal feasible solutions. It is easy to see why: while the sequence $\{\boldsymbol{v}(\boldsymbol{\pi}^\tau)\}$ generated from the dual subproblems converge to a simultaneously feasible and optimal solution, the feasible set in the problem (19) allows for large moves away from $\boldsymbol{v}(\boldsymbol{\pi}^\tau)$, and since the objective function is linear the scheme will produce extreme points of F that in general will not be optimal. In all fairness, Ventura developed his scheme in order to check whether the termination criterion (14) was a reasonable one, rather than as an optimizing one; however, the scheme's non-convergence may make the termination criterion unreliable in some cases.

3.2.2 Curet's scheme

In Curet's [Cur93] scheme, the size of any element of $\boldsymbol{\delta}$ is bounded by a constant $\varepsilon > 0$ which is also used in his dual algorithm, acting there as a relaxation parameter much like in ε-relaxation methods. His scheme ignores the objective function, and simply asks the vector $\boldsymbol{\delta}$ to be such that

$$\boldsymbol{E}(\boldsymbol{v}(\widehat{\boldsymbol{\pi}}) + \boldsymbol{\delta}) = \boldsymbol{d}, \tag{20a}$$

$$\max\{0, v_l(\widehat{\boldsymbol{\pi}}) - \varepsilon\} \leq v_l(\widehat{\boldsymbol{\pi}}) + \delta_l \leq \min\{u_l, v_l(\widehat{\boldsymbol{\pi}}) + \varepsilon\}, \qquad l \in \mathcal{L}. \tag{20b}$$

For any fixed value of $\varepsilon > 0$, it is obvious that the above system will become consistent eventually, as the value of $\boldsymbol{v}(\widehat{\boldsymbol{\pi}}) = \boldsymbol{v}(\boldsymbol{\pi}^\tau)$ converges to the primal optimal solution. Curet uses an a priori given value of $\varepsilon > 0$, and therefore provides a primal feasible solution finitely. (This solution also has a guaranteed quality in terms of the value of ϕ, measured in terms of the value of ε; see [Cur93, Proposition 3.3].) Curet's overall algorithm however has the drawback that there is a trade-off between getting fast convergence of the dual algorithm and primal feasible solutions quickly (from using large values of ε), and getting good primal solutions (which requires a small value of ε). We next consider a modification of his scheme which is projection-like when applied regularly from any convergent dual algorithm. In fact, it combines ideas both from Ventura's and Curet's schemes into one convergent scheme.

3.2.3 A modification of Curet's scheme

The *modified Curet* scheme is given by the following. We begin by using Ventura's scheme, to get a first primal feasible solution, denoted $P^V(\boldsymbol{v}(\widehat{\boldsymbol{\pi}}))$. We then calculate $\varepsilon := \max_{l \in \mathcal{L}} |(P^V(\boldsymbol{v}(\widehat{\boldsymbol{\pi}})))_l - v_l(\widehat{\boldsymbol{\pi}})|$. With this value of ε, the system (20) obviously is consistent. Moreover, the following linear single-commodity network flow modification of Ventura's scheme problem has $P^V(\boldsymbol{v}(\widehat{\boldsymbol{\pi}}))$ as an optimal solution:

$$\underset{\boldsymbol{\delta}}{\text{minimize}}\ \nabla\phi(\boldsymbol{v}(\widehat{\boldsymbol{\pi}}))^{\mathrm{T}}\boldsymbol{\delta}, \tag{21a}$$

$$\text{subject to } \boldsymbol{E}(\boldsymbol{v}(\widehat{\boldsymbol{\pi}}) + \boldsymbol{\delta}) = \boldsymbol{d}, \tag{21b}$$

$$\boldsymbol{v}(\widehat{\boldsymbol{\pi}}) + \boldsymbol{\delta} \in X, \tag{21c}$$

$$-\varepsilon \le \delta_l \le \varepsilon, \qquad l \in \mathcal{L}. \tag{21d}$$

The idea is to decrease the value of ε while ensuring the consistency of this system: we redefine $\varepsilon := \varepsilon/2$, and attempt to solve the problem (21) again, and thus keep reducing the value of ε until the above system becomes inconsistent, whence the last feasible solution found is the output of the scheme, denoted $P^{MC}(\boldsymbol{v}(\widehat{\boldsymbol{\pi}}))$.

Proposition 7 *The modified Curet scheme is projection-like.*

Proof. We have, for a given dual iterate $\boldsymbol{\pi}^\tau$,

$$\begin{aligned}\sum_{l \in \mathcal{L}} |(P^{MC}(\boldsymbol{v}(\boldsymbol{\pi}^\tau))_l - v_l(\boldsymbol{\pi}^\tau)| &= \sum_{l \in \mathcal{L}} |\delta_l^\tau| \le n\varepsilon_\tau \\ &< 2n \underset{\boldsymbol{v} \in F}{\text{minimum}}\ \underset{l \in \mathcal{L}}{\text{maximum}}\ |v_l - v_l(\boldsymbol{\pi}^\tau)| \\ &\le 2n \underset{l \in \mathcal{L}}{\text{maximum}}\ |v_l^* - v_l(\boldsymbol{\pi}^\tau)|.\end{aligned}$$

Here, we let $\boldsymbol{\delta}^\tau := P^{MC}(\boldsymbol{v}(\boldsymbol{\pi}^\tau)) - \boldsymbol{v}(\boldsymbol{\pi}^\tau)$ be the flow adjustment made in iteration τ, and we define ε_τ as the terminal value of ε in iteration τ; therefore, for the value $\varepsilon = \varepsilon_\tau/2$, we know that the problem (21) has no solution. In the above development, then, the first equality follows by definition, and the first inequality by the fact that $\boldsymbol{\delta}^\tau$ is feasible in (21). The next inequality stems from the fact that the maximal flow adjustment in any link must be greater than $\varepsilon_\tau/2$, by the definition of ε_τ. The last inequality then follows because $\boldsymbol{v}^*$ is a feasible solution to the problem in its left-hand side. As the last right-hand side tends to zero by the definition of a convergent dual algorithm, so does $\sum_{l \in \mathcal{L}} |(P^{MC}(\boldsymbol{v}(\boldsymbol{\pi}^\tau))_l - v_l(\boldsymbol{\pi}^\tau)|$. By the equivalence of norms, it tends to zero when τ tends to infinity if and only if $\|P^{MC}(\boldsymbol{v}(\boldsymbol{\pi}^\tau) - \boldsymbol{v}(\boldsymbol{\pi}^\tau)\|$ does, so we are done. $\square$

3.2.4 The minimum deviation scheme

After a change of variables the first of them, which will be referred to as the *minimum deviation* scheme, can be written as the result of solving the following linear program:

$$\underset{(\boldsymbol{\delta}^+, \boldsymbol{\delta}^-)}{\text{minimize}}\ (\mathbf{1}^n)^{\mathrm{T}}(\boldsymbol{\delta}^+ + \boldsymbol{\delta}^-), \tag{22a}$$

$$\text{subject to } \boldsymbol{E}(\boldsymbol{v}(\widehat{\boldsymbol{\pi}}) + \boldsymbol{\delta}^+ - \boldsymbol{\delta}^-) = \boldsymbol{d}, \tag{22b}$$

$$\boldsymbol{v}(\widehat{\boldsymbol{\pi}}) + \boldsymbol{\delta}^+ - \boldsymbol{\delta}^- \in X, \tag{22c}$$

$$\boldsymbol{\delta}^+, \boldsymbol{\delta}^- \ge \mathbf{0}^n. \tag{22d}$$

This is a linear single-commodity flow problem, where each link $l \in \mathcal{L}$ has two copies, one associated with an addition δ_l^+ of flow (that is, a link in the same direction as the original one), and the other with a subtraction δ_l^- of flow (and therefore being defined in the opposite direction). Note that while this is still a linear program, it does not have the same negative linear programming-based effects as Ventura's linear program: the objective function is constructed such that it is always non-negative, and the optimal value is zero if and only if the vector $\boldsymbol{v}(\widehat{\boldsymbol{\pi}}) \in F$.

PROPOSITION 8 *The minimum deviation scheme is projection-like.*

PROOF. Similarly to the proof for the modified Curet scheme, we have that

$$\begin{aligned}\sum_{l \in \mathcal{L}} |(P^{MD}(\boldsymbol{v}(\boldsymbol{\pi}^\tau)))_l - v_l(\boldsymbol{\pi}^\tau)| &= \underset{\boldsymbol{\delta}: \boldsymbol{v}(\boldsymbol{\pi}^\tau)+\boldsymbol{\delta} \in F}{\text{minimum}} \sum_{l \in \mathcal{L}} |\delta_l| \\ &\leq \sum_{l \in \mathcal{L}} |v_l^* - v_l(\boldsymbol{\pi}^\tau)|,\end{aligned}$$

where we know that the last right-hand side tends to zero as $\tau \to \infty$. Here, P^{MD} denotes the mapping that assigns a solution to the problem (22), and the last inequality holds because $\boldsymbol{\delta} := \boldsymbol{v}^* - \boldsymbol{v}(\boldsymbol{\pi}^\tau)$ is among the feasible solutions to the equivalent problem stated in the left-hand side of that inequality. $\square$

In the above result, we could also obtain an upper bound based on the fact that

$$\boldsymbol{E}\boldsymbol{\delta} = \boldsymbol{d} - \boldsymbol{E}\boldsymbol{v}(\boldsymbol{\pi}) = -\nabla\eta(\boldsymbol{\pi}) \tag{23}$$

holds for any adjustment vector $\boldsymbol{\delta}$ and $\boldsymbol{\pi}$. This identity will be used below.

In the following subsection we present four new projection-like schemes which all have in common that they are based on graph searches in the residual graph. The first is based on a breadth-first or depth-first search which provides flow imbalance-reducing paths. The second performs this task in a breadth search manner, in which case possibly several imbalance-reducing paths with the minimum number of links are produced. Finally, we present two optimizing graph search based schemes, both of which are based on shortest path searches. In contrast to the previous graph search based schemes they take the objective function of the problem (1) into account; the second of these is even devised to take into account the non-optimality of $\boldsymbol{v}(\boldsymbol{\pi}^\tau)$ in a clever way.

3.3 Graph search based realizations

3.3.1 The flowroute scheme

The second scheme uses the residual graph directly. We define source (sink) nodes to be those in the original graph where too much flow enters (respectively, too much flow leaves). This corresponds to nodes $i \in \mathcal{N}$ where $g_i(\boldsymbol{\pi}) := -\frac{\partial\eta(\boldsymbol{\pi})}{\partial\pi_i} > 0$ (respectively, $g_i < 0$). Since the total imbalance at $\boldsymbol{v}(\boldsymbol{\pi})$ is always identically zero [notice that, by (23), $\sum_{i \in \mathcal{N}} g_i(\boldsymbol{\pi}) = 0$], we can balance the network by sending flow along routes from the sources to the sinks. This is the goal of the *flowroute* scheme, which simultaneously complies with the requirement than $\boldsymbol{v}(\boldsymbol{\pi}) + \boldsymbol{\delta} \in X$ must hold.

PROPOSITION 9 *The flowroute scheme is projection-like.*

PROOF. We observe that since no more than the residual sources available can be rerouted, and no cycles can appear, the maximum adjustment on any particular link is the total residual supply, so we have that

$$\sum_{l \in \mathcal{L}} |(P^{FR}(\boldsymbol{v}(\boldsymbol{\pi}^\tau)))_l - v_l(\boldsymbol{\pi}^\tau)| \leq n \sum_{i \in \mathcal{N}} |g_i(\boldsymbol{\pi}^\tau)|, \tag{24}$$

where P^{FR} is the mapping defining the heuristic. As the algorithm converges, the right-hand side converges to zero, and we are done. $\square$

The flowroute scheme can be implemented either in a breadth-first or a depth-first manner; we denote them, respectively, by FlowrouteB (FB for short) and FlowrouteD (FD for short). From any given node the complexity of these schemes is $O(n + m)$.

3.3.2 The breadth search scheme

Breadth search is a graph search technique for traversing a capacitated network from a source node to a sink node. The result is an acyclic layered digraph (ALD) consisting of all routes leading from the origin to the destination defined by the minimal number of links. Its classic use is in polynomial max flow algorithms, while here we will apply it as a possible improvement of the flowroute scheme. The motivation is that breadth search in general yields more than one flow augmenting path through which we can reroute flow, and therefore a feasible flow may be found in much less iterations compared to the FlowrouteB scheme while they will probably generate similar routes. The complexity of breadth search is however higher: $O(n + m^2)$ for a given pair of origin and destination nodes.

To explain the scheme, based on the repeated construction of an ALD, in brief, each residual source (respectively, sink) is linked to a supersource (supersink) whose demand equals the total residual. In each iteration of the scheme we identify a node with the minimum capacity, defined as the least residual capacity of a connected link. Through this node we then route this flow from the supersource to the supersink along routes with the same (and minimal) number of links, that is, all the routes defined by the ALD, and in so doing we reduce the total residual. The algorithm terminates when the total residual reaches zero. An important note is that convergence of this scheme is only established for problems with integral data, which is not the case for the problems that we consider. Therefore, if it should terminate prematurely, we have implemented it such that the remaining imbalance is reduced by the use of the FlowrouteB scheme; see the following section.

Our notation for this scheme will be FlowrouteBS, or, in short, FBS.

PROPOSITION 10 *The breadth search scheme is projection-like.*

PROOF. The maximal amount of flow that can be rerouted is the sum of all the least node capacities; this is also the maximum amount of flow that can be rerouted through any one link. Hence, the bound in (24) applies here as well (of course with F^{FR} replaced by F^{FBS}), and we are done. ☐

3.3.3 The shortest path scheme

The *shortest path* scheme is similar to flowroute; routes are here however selected to be the shortest with respect to the linearized cost $\boldsymbol{\delta} \mapsto \nabla\phi(\boldsymbol{v}(\boldsymbol{\pi}^\tau))^{\mathrm{T}}\boldsymbol{\delta}$. We denote the scheme by Shortest Path (or, for short, SP).

PROPOSITION 11 *The shortest path scheme is projection-like.*

PROOF. In order to establish that this method is feasible, we must show that there are no negative cycles in the residual graph with respect to the cost function chosen. This is easily established, based on Everett's Theorem ([Eve63]). Applied to the Lagrangian subproblem (3) it implies that the vector $\boldsymbol{v}(\boldsymbol{\pi}^\tau)$ is an optimal solution to the problem of minimizing $\phi(\boldsymbol{v})$ over the constraints that $\boldsymbol{E}\boldsymbol{v} = \boldsymbol{E}\boldsymbol{v}(\boldsymbol{\pi}^\tau)$ and $\boldsymbol{v} \in X$ hold. The first-order optimality conditions for this problem state that $\nabla\phi(\boldsymbol{v}(\boldsymbol{\pi}^\tau))^{\mathrm{T}}\boldsymbol{\delta} \geq 0$ must hold for every feasible direction $\boldsymbol{\delta}$ with respect to F at $\boldsymbol{v}(\boldsymbol{\pi}^\tau)$, which implies the result sought; cf. Proposition 3.

We can also establish that the scheme is projection-like, by using arguments similar to that for the flowroute scheme. As we do not reroute more flow than is available in the residual graph, and no cycles appear in the updates due to what was just established, the bound in proof of Proposition 9 applies. ☐

3.3.4 A shortest path scheme based on Lagrangian costs

In this scheme we replace the linear cost vector $\boldsymbol{\delta} \mapsto \nabla\phi(\boldsymbol{v}(\boldsymbol{\pi}^\tau))^{\mathrm{T}}\boldsymbol{\delta}$ in the shortest path scheme of Section 3.3.3 by the linearization of the Lagrange function:

$$\boldsymbol{\delta} \mapsto [\nabla\phi(\boldsymbol{v}(\boldsymbol{\pi}^\tau)) + \boldsymbol{E}^{\mathrm{T}}\boldsymbol{\pi}^\tau]^{\mathrm{T}}\boldsymbol{\delta}.$$

The motivation for this choice stems from the global optimality conditions of the primal problem, to be discussed below once we have established the following.

PROPOSITION 12 *The Lagrangian shortest path scheme is projection-like.*

Proof. We first establish that the scheme is well-defined. At a given iteration τ the vector $\boldsymbol{v}(\boldsymbol{\pi}^\tau)$ is a globally optimal solution to the problem of minimizing $\phi(\boldsymbol{v}) + (\boldsymbol{\pi}^\tau)^{\mathrm{T}}(\boldsymbol{E}\boldsymbol{v} - \boldsymbol{d})$ over $\boldsymbol{v} \in X$. Its optimality conditions state that for every feasible direction $\boldsymbol{\delta}$ of X it holds that $[\nabla\phi(\boldsymbol{v}(\boldsymbol{\pi}^\tau)) + \boldsymbol{E}^{\mathrm{T}}\boldsymbol{\pi}^\tau]^{\mathrm{T}}\boldsymbol{\delta} \geq 0$. Hence, no negative cycles can appear in the shortest path problem having the given link costs, and so the scheme will provide an optimal solution which is feasible in the original problem.

The remainder of the proof is identical to that of Proposition 11 for the shortest path scheme. □

A short motivation for the above scheme, to be denoted Shortest PathL or SPL, is given next. Its basis is a theory of global optimality and global near-optimality in [LaP06]. Suppose that the dual vector $\boldsymbol{\pi}^\tau$ is β-optimal in the dual problem (7), that is, that $\eta(\boldsymbol{\pi}^\tau) \geq \eta^* - \beta$, for some $\beta \geq 0$. Suppose also that the vector $\hat{\boldsymbol{v}}$ is a feasible solution in the primal problem (1) which is ε-optimal in the Lagrangian minimization problem in (3), that is, that $\phi(\hat{\boldsymbol{v}}) + (\boldsymbol{\pi}^\tau)^{\mathrm{T}}(\boldsymbol{E}\hat{\boldsymbol{v}} - \boldsymbol{d}) \leq \eta(\boldsymbol{\pi}^\tau) + \varepsilon$ holds for some $\varepsilon \geq 0$. The vector $\hat{\boldsymbol{v}}$ can be seen as a vector that has been generated by some primal feasibility scheme, starting from the Lagrangian minimum $\boldsymbol{v}(\boldsymbol{\pi}^\tau)$; in the process, of course, the vector loses optimality in the Lagrangian problem, as measured by the value of ε. Provided that $\varepsilon \geq \beta$ holds, $\hat{\boldsymbol{v}}$ is shown in [LaP06] to be $(\varepsilon - \beta)$-optimal in the problem (1). In particular, we see that if the value of β is small then the adjustment from $\boldsymbol{v}(\boldsymbol{\pi}^\tau)$ towards a feasible solution through a primal feasibility scheme should not increase the value of the Lagrange function too much. Its linearization is used in the Lagrangian shortest path scheme in order to try to comply with this requirement.

4 Numerical experiments

The feasibility schemes presented above have all been implemented and tested, through two masters projects supervised by two of the authors. The first masters project, performed by Marklund [Mar93], concentrated on non-optimizing schemes, and included the derivation of the modified Curet and minimum deviation schemes as well as the flowroute scheme. The second masters project, performed by Olsson [Ols05], concentrated on optimizing schemes, and included the derivation of the breadth search scheme as well as the two shortest path schemes.

The two masters theses contain numerical experiments with dense and sparse quadratic transportation problem instances generated by the principles in [OhK84]; Olsson's thesis includes in addition experiments done on quadratic transshipment problems generated from NETGEN [KNS74]. Rather than repeat all these experiments, this section summarizes results on the most interesting feasibility schemes. To this end, Marklund's experiments are particularly interesting, as they led to the conclusion that all schemes *not* based on graph searches most be disqualified. The experiments performed in Olsson [Ols05] therefore concerned the following schemes: FlowrouteB, FlowrouteD, FlowrouteBS, ShortestPath, and ShortestPathL. The following discussion relates to these latter experiments. We remark that while we provide results only for one random number seed, the results presented are representative of a larger set of instances that we also have tested.

4.1 Transportation network instances and algorithmic coding

The system described in [OhK84] generates complete networks with strictly convex quadratic and separable costs. Parameters that must be pre-selected include the number of sources (s) and sinks (t), the average amount of flow on a link ($h \in (0, 1)$; a low (high) value gives an optimal solution whose total link flow is far from (close to) the total demand), maximum values for the cost coefficients (a [linear terms], b [quadratic terms]), and a random number seed (here taken to be 13502460, as recommended in [KNS74]). We generated 12 instances of complete graphs and as many sparse graphs (with 10 % of the links present), with parameter values set according to Table 1.

Note that we create pairs of problems with different degrees of cost nonlinearities, even numbered instances being relatively more nonlinear. Further, examples 1–6 have a total capacity far from the total demand, while examples 7–12 have a total capacity close to the total demand.

As noted previously, our choice of dual algorithm is the conjugate gradient algorithm using the Polak–Ribiere–Polyak formula (12). The line search was taken to be an exact one, solved by using a sorting of the break points of the derivative of the one-dimensional dual function. The sorting itself was performed using Shellsort, which has a quadratic worst-case complexity but also re-sorts easily. As a stopping criterion for the dual algorithm we have used a relative objective value tolerance based on a pre-calculated

Ex	s	t	h	a	b
1	50	50	0.3	10	1
2	50	50	0.3	1	10
3	100	100	0.3	10	1
4	100	100	0.3	1	10
5	100	200	0.3	10	1
6	100	200	0.3	1	10
7	50	50	0.7	10	1
8	50	50	0.7	1	10
9	100	100	0.7	10	1
10	100	100	0.7	1	10
11	100	200	0.7	10	1
12	100	200	0.7	1	10

Table 1: Selection of parameter values for transportation network generation

near-optimal primal feasible solution (satisfying $\|\nabla\eta(\boldsymbol{\pi}^\tau)\|/\|\boldsymbol{d}\| \leq 1e^{-12}$)), which we refer to as $\boldsymbol{v}^*$; the tolerance measure is

$$\Delta_\tau := 100\frac{\phi(\boldsymbol{v}^*) - \eta(\boldsymbol{\pi}^\tau)}{\phi(\boldsymbol{v}^*)}. \tag{25}$$

We have as stopping criterion that $\Delta_\tau \leq \delta$, where δ has been chosen as 0.1% or 0.001% (which will be noted for each test case reported).

4.2 Numerical results

We will report upon numerical results in two ways. We provide a measure of the quality of the primal schemes in terms of the primal objective value reached in the final iteration. For a given primal scheme, whose "projection" operation is P, this quality measure is

$$Q_P := 100\frac{\phi(P(\boldsymbol{v}(\boldsymbol{\pi}^\tau)) - \phi(\boldsymbol{v}^*)}{\phi(\boldsymbol{v}^*)}. \tag{26}$$

4.2.1 Dense transportation networks

Our first tables of results show the relative quality of the flowroute schemes (using breadth-first [FB] (Section 3.3.1), breadth [FBS] (Section 3.3.2), or depth-first [FD] (Section 3.3.1) search) and the shortest path schemes (using linearized original link costs [SP] (Section 3.3.3) or linearized Lagrangian link costs [SPL] (Section 3.3.4)). In Table 2 the quality of the best scheme is marked with **bold** text, while the worst is marked with *slanted* text. The two tables correspond to using the two stopping criteria $\Delta_\tau \leq \delta$ where $\delta = 0.1\%$ and $\delta = 0.001\%$, respectively.

The reader should observe in the following that the breadth-search scheme (FBS) has an inherent problem in that it may fail to provide flow-augmenting routes even if an imbalance still exists; this however has only happened in practice when the network is already nearly balanced. (As remarked upon earlier, theoretically convergence of the FBS scheme has only been established for integral data, which is not our situation.) In order to produce a feasible solution even when breadth-search fails we have implemented it such that when this problem occurs, a continuation utilizes breadth-first search. Throughout the tables below where we report our tests, an asterisk '*' denotes a case where this problem has occurred, and the corresponding data in the tables represent the total time it took to provide the resulting feasible solution, and its quality.

We next, in Table 3, record the CPU times corresponding to the above two cases.

To illustrate the relative performances of our heuristics we study their convergence when the stopping criterion for the conjugate gradient method was "$\delta \leq 10^{-7}\%$" (which occurs at $\tau = 20$) for the network in Example 1. Figure 1 shows the convergence of the primal objective value given by the different schemes vs. the dual objective value given by the dual algorithm.

Figure 2 further illustrates for the same problem the convergence similarities between some of the schemes, plotting the quality measure $Q_P/100$ against the dual quality measure $\Delta_\tau/100$, cf. (26) and (25), respectively.

Ex	Q_{FB}	Q_{FBS}	Q_{FD}	Q_{SP}	Q_{SPL}
1	*1.077*	0.606*	0.997	0.387	**0.374**
2	0.869	0.761*	*0.896*	**0.593**	0.664
3	1.004	0.467*	*1.815*	0.377	**0.347**
4	1.375	0.743*	*2.393*	0.612	**0.601**
5	2.078	0.576*	*2.380*	0.625	**0.622**
6	*0.562*	0.388*	0.524	**0.305**	0.317
7	*1.535*	**0.282***	0.549	0.499	0.532
8	*1.776*	**0.100***	0.641	0.772	0.620
9	*0.828*	**0.109***	0.498	0.338	0.348
10	*1.000*	**0.190***	0.758	0.630	0.424
11	*1.464*	**0.314***	1.291	0.788	0.820
12	*1.777*	**0.423***	1.510	1.201	1.102

Ex	Q_{FB}	Q_{FBS}	Q_{FD}	Q_{SP}	Q_{SPL}
1	*0.0840*	0.0625*	0.0757	**0.0447**	0.0460
2	*0.1687*	0.0824*	0.1388	**0.0571**	0.0580
3	*0.1004*	0.0777*	0.0756	0.0198	**0.0169**
4	*0.1332*	0.0943*	0.1305	0.0294	**0.0240**
5	*0.0907*	0.0658*	0.0825	**0.0425**	0.0439
6	*0.1262*	0.0964	0.1149	**0.0748**	0.0750
7	*0.1597*	0.0661*	0.1070	**0.0558**	0.0593
8	*0.1709*	0.0800	0.0977	0.0839	**0.0631**
9	*0.0641*	0.0272*	0.0346	0.0220	**0.0219**
10	*0.0730*	0.0317*	0.0369	0.0389	**0.0256**
11	*0.1086*	0.0390*	0.0575	0.0372	**0.0361**
12	*0.1358*	0.0522*	0.0671	0.0478	**0.0466**

Table 2: Dense transportation problems. Quality (value of Q_P) of feasible solutions for $\delta = 0.1\%$ (left table) respectively $\delta = 0.001\%$ (right table). In FBS, a '*' denotes that FB is used to reroute on average 13% (left table) respectively 1 % (right table) of the initial flow imbalance.

Ex	T_{FB}	T_{FBS}	T_{FD}	T_{SP}	T_{SPL}
1	156	2632*	**94**	5515	*7082*
2	161	3034*	**97**	6844	*7555*
3	193	4175*	**128**	11,448	*14,787*
4	214	4153*	**103**	11,134	*14,128*
5	280	6848*	**152**	18,796	*23,864*
6	326	7730*	**119**	23,000	*28,170*
7	184	1988*	**61**	4741	*6087*
8	157	1754*	**63**	4824	*6547*
9	212	3505*	**83**	9534	*11,876*
10	196	3422*	**90**	9664	*12,544*
11	309	5092*	**128**	15,678	*19,246*
12	244	5378*	**126**	14,653	*17,845*

Ex	T_{FB}	T_{FBS}	T_{FD}	T_{SP}	T_{SPL}
1	125	2569*	**63**	5389	*6925*
2	129	2421*	**65**	6038	*7006*
3	169	6133*	**72**	11,793	*14,852*
4	238	5924*	**95**	11,714	*14,835*
5	331	9077*	**92**	24,431	*30,840*
6	325	8612	**95**	23,735	*29,871*
7	92	2080*	**61**	4344	*5506*
8	94	412	**63**	4761	*5795*
9	227	3574*	**60**	10,357	*12,941*
10	196	3634*	**90**	9874	*12,642*
11	263	5280*	**113**	14,203	*17,715*
12	251	5474*	**78**	13,115	*16,362*

Table 3: Dense transportation problems. Running times (in seconds) for $\delta = 0.1\%$ (left table) respectively $\delta = 0.001\%$ (right table).

4.2.2 Sparse transportation networks

For the case of sparse networks, where the number of links is 10% of the total number of possible ones, we generated information similar to that for the previous section. In Table 4 we record the relative solution quality of the different schemes for the cases of $\delta \leq 0.1$ and $\delta \leq 0.001$, respectively.

Ex	Q_{FB}	Q_{FBS}	Q_{FD}	Q_{SP}	Q_{SPL}
1	3.430	0.899	*4.736*	**0.736**	0.785
2	2.264	0.806	*7.188*	**0.413**	0.486
3	1.973	0.684*	*3.658*	0.482	**0.434**
4	2.103	0.795*	*4.474*	0.342	**0.297**
5	1.849	0.691*	*4.918*	**0.345**	0.377
6	1.865	0.607*	*8.338*	0.357	**0.303**
7	2.183	0.817*	*5.379*	0.745	**0.718**
8	2.648	0.857*	*8.640*	0.765	**0.690**
9	2.209	0.903	*6.068*	0.821	**0.800**
10	2.523	**1.114***	*6.540*	1.198	1.127
11	1.751	0.651*	*4.829*	**0.600**	0.615
12	1.915	0.776*	*5.335*	0.774	**0.666**

Ex	Q_{FB}	Q_{FBS}	Q_{FD}	Q_{SP}	Q_{SPL}
1	0.1542	0.0785*	*0.2832*	**0.0605**	0.0625
2	0.1428	0.0986*	*0.3315*	0.0612	**0.0589**
3	0.1982	0.0781	*0.3967*	**0.0596**	0.0664
4	0.2567	0.0608	*0.4283*	**0.0322**	0.0335
5	0.1807	0.0563*	*0.5656*	**0.0263**	0.0333
6	0.1701	0.0555*	*0.5936*	0.0249	**0.0236**
7	0.0719	**0.0210**	*0.2709*	0.0364	0.0368
8	0.1222	**0.0420**	*0.3964*	0.0686	0.0638
9	0.1347	0.0637*	*0.3130*	**0.0561**	0.0627
10	0.1098	**0.0416**	*0.3184*	0.0507	0.0464
11	0.1224	0.0547*	*0.5408*	**0.0501**	0.0522
12	0.1285	0.0607*	*0.6232*	0.0649	**0.0597**

Table 4: Sparse transportation problems. Quality (value of Q_P) of feasible solutions for $\delta = 0.1\%$ (left table) respectively $\delta = 0.001\%$ (right table). In FBS, a '*' denotes that FB is used to reroute on average $< 1\%$ (left table) respectively 1% (right table) of the initial flow imbalance.

We next, in Table 5, record the CPU times corresponding to the above two cases.

To illustrate the relative performances of our heuristics we study their convergence when the stopping criterion for the conjugate gradient method was $\delta \leq 10^{-7}\%$ (which occurs at $\tau = 228$) for the network in Example 1. Figure 3 shows the convergence of the primal objective value given by the different schemes vs. the dual objective value given by the dual algorithm.

Figure 4 further illustrates the convergence similarities between some of the schemes, plotting the

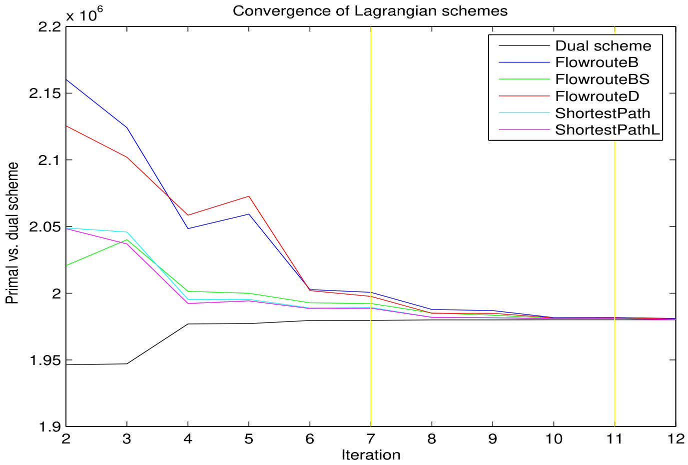

Figure 1: Dense transportation problems. Example 1. Convergence towards $\phi^* \approx 1980065$ of the primal objective value given by the different schemes and the dual objective value given by the dual algorithm. The case of $\delta = 0.1\%$ ($\delta = 0.001\%$) is shown at $\tau = 7$ ($\tau = 11$).

Ex	T_{FB}	T_{FBS}	T_{FD}	T_{SP}	T_{SPL}
1	425	5307	**212**	4882	_5519_
2	432	6053	**216**	7567	_8648_
3	532	10,045*	**295**	9749	_11,877_
4	530	10,299*	**471**	11,594	_13,771_
5	583	15,284*	**522**	17,494	_21,269_
6	**574**	14,539*	725	17,532	_21,522_
7	**212**	_6144*_	424	3814	4449
8	**407**	_5089*_	**407**	4885	_5089_
9	**493**	8214	712	8324	_9748_
10	**592**	8875*	**592**	9036	_11,672_
11	710	12,097*	**526**	13,491	_16,831_
12	812	13,163*	**348**	15,714	_18,556_

Ex	T_{FB}	T_{FBS}	T_{FD}	T_{SP}	T_{SPL}
1	637	5519*	**212**	_5732_	4033
2	**216**	_5837*_	**216**	3675	3891
3	591	9454	**414**	10,813	_14,299_
4	647	8769	**294**	13,830	_17,891_
5	645	14,947*	**583**	17,248	_21,545_
6	665	14,872*	**302**	16,383	_19,859_
7	424	4449	**212**	4025	_5085_
8	814	5293	**407**	4885	_6107_
9	548	8379*	**383**	7995	_10,952_
10	592	7853	**323**	8391	_11,134_
11	605	11,781*	**500**	13,622	_15,700_
12	783	14,294*	**406**	15,569	_19,628_

Table 5: Sparse transportation problems. Running times (in seconds) for $\delta = 0.1\%$ (left table) respectively $\delta = 0.001\%$ (right table).

quality measure $Q_P/100$ against the dual quality measure $\Delta_\tau/100$, cf. (26) and (25), respectively.

4.2.3 Discussion

FlowrouteD vs. FlowrouteB For the *dense* networks, FlowrouteD produces results about twice as fast as FlowrouteB on average. In terms of quality, at $\delta = 0.1\%$, FlowrouteD wins in all the more congested cases (Ex. 7–12) but FlowrouteB wins in 4 out of the 6 less congested ones; at $\delta = 0.001\%$, FlowrouteD wins in all cases.

For the *sparse* networks, FlowrouteD produces results 1.5 times faster than FlowrouteB on average. The running time was shorter for 8 out of 12 of the more nonlinear problems. In terms of quality, FlowrouteB wins in all cases.

We believe that the fact that FlowrouteD produces results of lower quality than FlowrouteB for sparse problems can be explained by FlowrouteD having problems finding residual sinks when DFS is truncated prematurely.

FlowrouteBS vs. FlowrouteB For the *dense* networks, FlowrouteBS produces results 25 times slower than FlowrouteB on average. Further, FlowrouteBS terminated prematurely in all but 2 of the 24 cases. However, FlowrouteBS produces results of significantly better quality, especially for the more linear problems; in 5 out of the 6 more nonlinear problems in Ex. 1–6, and in all of the more linear problems Ex. 7–12.

For the *sparse* networks, FlowrouteBS produces results 20 times slower than FlowrouteB on average.

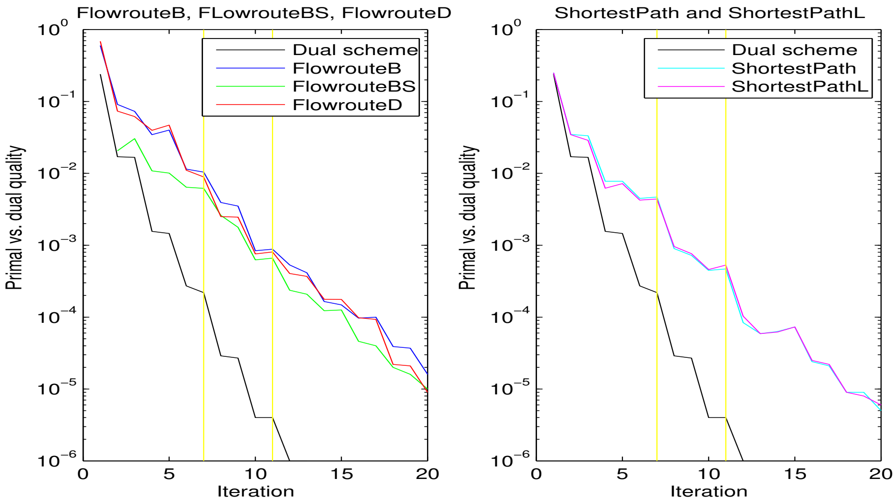

Figure 2: Dense transportation problems. Example 1. Quality measures for the dual sequence and the different schemes in logarithmic scale. The left figure shows three flowroute schemes; the right figure shows two shortest path schemes.

Further, FlowrouteBS terminated prematurely in 4 out of these 6 cases. However, FlowrouteBS produces results of significantly better quality, especially for the more linear problems; in 4 out of the 6 more nonlinear problems in Ex. 1–6, and in 5 out of the 6 more linear problems in Ex. 7–12.

We believe that the reason for the success of FlowrouteBS in producing results of such high quality is that the trees of minimum path length in terms of the number of links almost always contains the minimum cost paths from the supersource to the supersink (the number of links and the total cost of a route are of course two highly correlated values), and that using fewer arcs to reroute flow is also favourable in quality terms. However, as reported earlier, we found that the supersink of the (on average) third to fifth ALD created by the heuristic could not be reached from the supersource, which is crucial for the algorithm.

Shortest Path vs. Shortest PathL For the *dense* networks, Shortest Path produces results 1.3 times faster than Shortest PathL on average. The running time was not affected by the linearity of the problem. In terms of solution quality, Shortest PathL is better in 14 of the 24 cases, with the results for $\delta \leq 0.1$ having better quality in 4 out of Ex. 1–6, and in 3 out of Ex. 7–12; for $\delta \leq 0.001$ having better quality in 2 out of Ex. 1–6, and in 5 out of Ex. 7–12.

For the *sparse* networks; Shortest PathL produces results 1.3 times slower than Shortest Path on average. In terms of solution quality, Shortest PathL is better only in 10 of the 24 cases, but it wins in 7 out of the 12 more nonlinear cases; relatively speaking Shortest PathL also fares better for the case of $\delta \leq 0.1$ (7 wins out of 12) than for the case of $\delta \leq 0.001$ (3 wins out of 12).

The longer running times in Shortest PathL are explained by the extra computations needed to calculate the derivative of the Lagrangian cost function.

We also note that FlowrouteBS compares quite favourably with the two shortest path schemes, especially so for the more dense and linear transportation problems.

4.3 Transshipment problems

The tests for transshipment networks generated by the network generator NETGEN ([KNS74]) were conducted on 12 networks. The parameters to NETGEN were set similarly as in the standard examples 16–19 and 24–27 described in [KNS74], with the number of vertices equal to 400; the number of transshipment sources and transshipment sinks equal to zero; the total supply equal to 400000; the parameter *highcost* equal to 30; the parameter *capacitated* equal to 20; and the seed to the random number generator equal to 13502460. In its original version, NETGEN creates problems with linear cost coefficients only, but we modified it to create problems with quadratic cost coefficients as well.

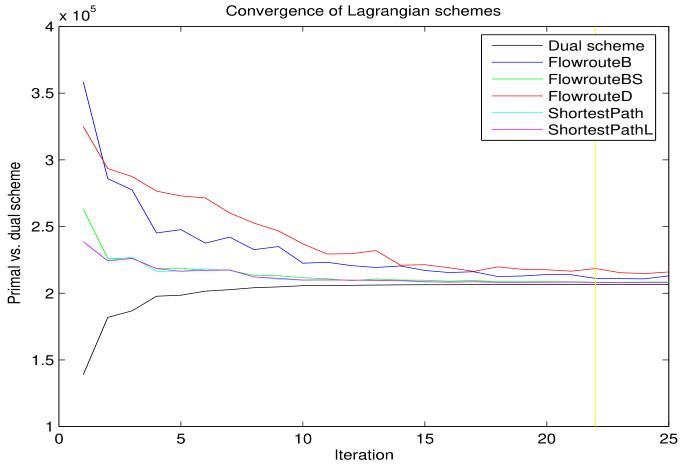

Figure 3: Sparse transportation problems. Example 1. Convergence towards $\phi^* \approx 206737$ of the primal objective value given by the different schemes and the dual objective value given by the dual algorithm. The case of $\delta = 0.1\%$ ($\delta = 0.001\%$) is shown at $\tau = 22$ ($\tau = 228$).

Table 6 summarizes the instances produced.

Ex	*v*	*s*	*t*	*density*	*a*	*b*	*supply*	*h_cost*	*cap*	*mincap*	*maxcap*
1	400	8	60	1306	10	1	400,000	30	20	16000	30000
2	400	8	60	1306	5	10	400,000	30	20	16000	30000
3	400	8	60	2443	10	1	400,000	30	20	16000	30000
4	400	8	60	2443	5	10	400,000	30	20	16000	30000
5	400	8	60	6,000	10	1	400,000	30	20	16000	30000
6	400	8	60	6,000	5	10	400,000	30	20	16000	30000
7	400	8	60	1306	10	1	400,000	30	20	20000	120000
8	400	8	60	1306	5	10	400,000	30	20	20000	120000
9	400	8	60	2443	10	1	400,000	30	20	20000	120000
10	400	8	60	2443	5	10	400,000	30	20	20000	120000
11	400	8	60	6,000	10	1	400,000	30	20	20000	120000
12	400	8	60	6,000	5	10	400,000	30	20	20000	120000

Table 6: Selection of parameter values for transshipment network generation

In Table 7 we record the relative qualities of the primal schemes proposed, in the same manner as for the transportation problem instances, and for the same termination criteria for the dual problem.

We next, in Table 8, record the CPU times corresponding to the above two cases.

4.4 Discussion

FlowrouteD and FlowrouteB vs. FlowrouteBS For the transshipment problems FlowrouteB and FlowrouteD produce results up to 70 times faster than FlowrouteBS. In FlowrouteB the running time was not affected by the linearity of the problem on average, and it is generally the fastest, in 10 out of the 12 cases for $\delta \leq 0.1\%$, and in 8 out of the 12 cases for $\delta \leq 0.001\%$.

However, FlowrouteBS produced results with up to 35 times the quality of FlowrouteD, and with up to 15 times the quality of FlowrouteB. It also provides better solutions than both Shortest Path and Shortest PathL in 8 out of the 24 cases, however in most cases with a rather small margin.

FlowrouteD vs. FlowrouteB For the more *dense* transshipment problems (ex. 5, 6, 11, and 12) FlowrouteD produces results up to 5 times faster than, and not worse than twice the quality of FlowrouteB.

For the more *sparse* transshipment problems (ex. 1, 2, 7, and 8) FlowrouteB produces results with up to 10 times the quality of, and up to 4 times faster than FlowrouteD. FlowrouteD faired comparatively better on more nonlinear problems.

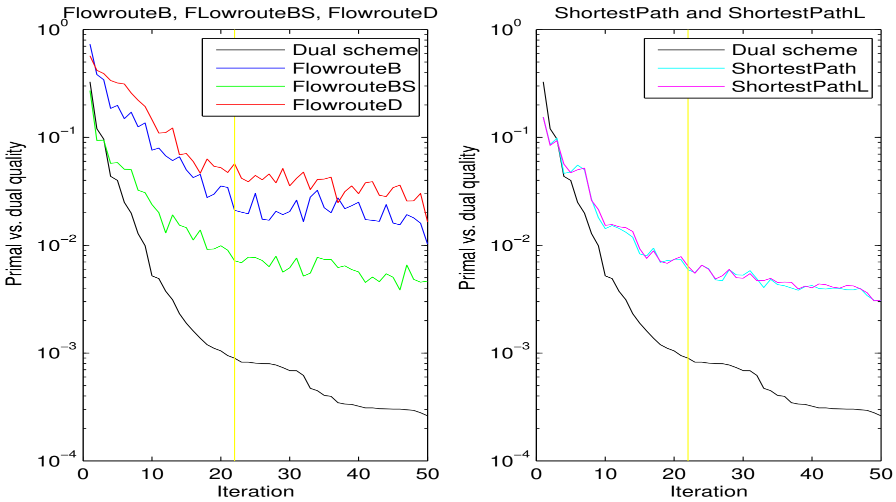

Figure 4: Sparse transportation problems. Example 1. Quality measures for the dual sequence and the different schemes in logarithmic scale. Left figure shows three flowroute schemes; right figure shows two shortest path schemes.

Ex	Q_{FB}	Q_{FBS}	Q_{FD}	Q_{SP}	Q_{SPL}
1	0.363	0.256*	*3.078*	**0.192**	0.195
2	0.878	0.372*	*6.171*	**0.178**	0.185
3	1.436	0.143*	*2.151*	**0.073**	0.163
4	0.998	**0.195***	*2.536*	0.382	0.309
5	1.504	0.651*	*1.579*	0.698	**0.353**
6	*5.940*	**0.462***	1.119	0.465	2.917
7	0.501	0.354*	*4.394*	**0.049**	0.220
8	0.498	0.347*	*4.451*	**0.100**	0.105
9	1.192	0.183*	*2.364*	**0.106**	0.148
10	0.846	0.139*	*2.301*	0.115	**0.086**
11	1.504	0.651*	*1.579*	0.698	**0.353**
12	*5.940*	**0.462***	1.119	0.465	2.917

Ex	Q_{FB}	Q_{FBS}	Q_{FD}	Q_{SP}	Q_{SPL}
1	0.0365	0.0229*	*0.2728*	**0.0183**	0.0186
2	0.0680	0.0322*	*0.4343*	0.0231	**0.0227**
3	0.0324	0.0041	*0.1464*	0.0036	**0.0034**
4	0.0265	0.0042	*0.1504*	**0.0031**	0.0083
5	0.0570	0.0256	*0.0602*	**0.0228**	0.0257
6	0.0383	**0.0225**	*0.0605*	0.0226	0.0303
7	0.0474	**0.0161***	*0.3698*	0.0244	0.0247
8	0.0496	**0.0162***	*0.3694*	0.0245	0.0249
9	0.0738	0.0293*	*0.3106*	**0.0267**	0.0343
10	0.0804	**0.0109***	*0.1650*	0.0159	0.0339
11	0.0570	0.0256*	*0.0602*	**0.0228**	0.0257
12	0.0383	**0.0225**	*0.0605*	0.0226	0.0303

Table 7: Transshipment problems. Quality of feasible solutions for $\delta = 0.1\%$ (left table) respectively $\delta = 0.001\%$ (right table). In FBS, a '*' denotes that FlowrouteB is used to reroute on average 1% (left table) respectively 5% (right table) of the initial flow imbalance.

Shortest Path vs. Shortest PathL For $\delta = 0.1$ Shortest Path produces results of a better quality in most cases; Shortest PathL however fares relatively better when utilized far from an optimal dual vector. Shortest Path produces results 1.2 times faster than Shortest PathL on average.

4.5 Conclusions from the experiments

The generally fastest schemes are FlowrouteD and FlowrouteB, and they are also generally the ones that provide the worst feasible solutions among the graph search based methods tested. (Further, the generally best schemes in terms of solution quality are Shortest Path and Shortest PathL, followed by FlowrouteBS, and they are also generally the slowest.

There are also other trends that can be found in the data provided from the above experiments:

- *Quality:* In a relative sense, Shortest PathL provides better feasible solutions than Shortest Path especially at dual solutions that are further from the optimum; also, Shortest PathL generally provides better solutions when the problem is more nonlinear. Both these findings can be explained theoretically: Shortest PathL is designed to take into account the level of non-optimality of the dual vector through the additional link cost term, which also compensates to a better degree for the fact that the shortest path problem is linear and therefore provides an extreme solution. The difference between Shortest Path and Shortest PathL is, however, most often quite small, so they

Ex	T_{FB}	T_{FBS}	T_{FD}	T_{SP}	T_{SPL}
1	**554**	*34,698**	2215	12,504	12,781
2	**1036**	*36,181**	1932	19,551	20,776
3	**566**	*22,701**	850	15,332	18,053
4	**536**	*23,531**	931	15,433	17,775
5	**193**	7477*	206	6826	*8352*
6	193	6866*	**131**	7609	*9135*
7	**863**	*40,193**	1817	14,488	15,033
8	**1159**	*35,018**	2498	9511	9271
9	**462**	*14,730**	493	10,035	12,683
10	**319**	*13,191**	425	9290	10,216
11	**364**	14,520*	**364**	13,260	*16,546*
12	411	15,158*	**270**	16,353	*19,627*

Ex	T_{FB}	T_{FBS}	T_{FD}	T_{SP}	T_{SPL}
1	**600**	*32,299**	2815	11,674	12,550
2	**1036**	*41,128**	2732	9422	9893
3	**283**	*22,474*	538	9324	11,308
4	**310**	*21,979*	621	10,411	12,048
5	142	*6710*	**71**	5590	6408
6	**68**	*4812*	74	2451	2860
7	**772**	*24,661**	2725	15,396	18,121
8	**769**	*20,079**	2258	17,389	19,358
9	462	*14,130**	**308**	10,928	13,191
10	**197**	*10,960**	440	6330	6937
11	311	*13,001**	**156**	10,455	12,598
12	167	*9820*	**128**	5186	6084

Table 8: Transshipment problems. Running time (in seconds) for $\delta = 0.1\%$ (left table) respectively $\delta = 0.001\%$ (right table).

are in effect quite comparable in terms of solution quality.

- *Quality:* FlowrouteBS is designed to provide solutions given by flows on routes that use few links. When, in particular, solving network flow problems over bipartite graphs, our experience is that FlowrouteBS provides, in a relative sense compared to the Shortest Path and Shortest PartL schemes, better solutions when the flows are near-capacitated. In almost all cases it fares almost as well as the other two, and even wins in some cases even though it is not based on any optimization; it always wins over the simpler FlowrouteB and FlowrouteD schemes. (An argument for the former is that due to the nonlinearity of the link costs, it may be better to adjust the flow on more (and relatively short in terms of number of links) paths than on fewer (and relatively long) paths.) FlowrouteD, in particular, almost always provides the worst solutions., and it is only for the dense bipartite problems where FlowrouteB is worse. To some degree, the bad results for FlowrouteD can be explained by the fact that the routes that it produces in general include more links than those generated by FlowrouteB and in particular compared to FlowrouteBS; the cost of a route is highly correlated with the number of links defining it, which implies that the costs of the routes produced by FlowrouteD often are higher.
- *Time:* For bipartite graphs FlowrouteD is always the fastest, followed by FlowrouteB. For these problems, Shortest PathL is always the slowest, followed by Shortest Path. The latter can, to some degree, be explained by the extra calculations involved in Shortest PathL when determining the link costs. For transshipment problems, FlowrouteB generally is the fastest; in some cases, FlowrouteD is equally fast. FlowrouteBS is always the slowest among the schemes.
- *Time:* Comparing the running times for the primal feasibility schemes between the less and more accurate dual starting points, no general conclusion can be made. This is however natural: although there is much less flow to be rerouted in the latter case, most of the nodes are still unbalanced.

5 Conclusions and recommendations

From the above experiments we conclude that while the shortest path based schemes in general provide primal solutions of the best quality they are also the slowest ones; vice versa, the simplest and most efficient graph search methods FlowrouteD and FlowrouteB provide primal flows of the worst quality.

We last return to the discussion in the first section about the utilization of Lagrangian heuristics in practice, and form the following general recommendation: If one applies a Lagrangian heuristic with the objective to find a rough primal feasible solution, then one should apply the simplest and fastest graph search methods, FlowrouteB or FlowrouteD; they are not based on any optimization but they are very fast. If one, close to an optimal solution, wishes to obtain a good quality primal feasible flow for further analysis, or for use in a stopping criterion, then one would only apply a Lagrangian heuristic once or a few times only, whence its computational complexity is less crucial but on the other hand the quality of the feasible solution is; in this case either of the two optimization based Shortest Path or Shortest PathL schemes is recommended.

Our general conclusion is that for the nonlinear network flow problem primal feasibility schemes are quite effective in providing feasible flows from a dual solution. Further, the quality of these primal flows may be comparable to those of the dual solution itself.

It is possible to extend these types of procedures also to other linearly constrained optimization problems, provided of course that the structure of the feasible set enables the construction of feasibility-inducing manipulative searches in the null space of the constraint matrix.

Acknowledgments

We thank the editor Roman Slowinski and the three anonymous reviewers for constructive criticism which allowed us to improve upon the presentation of the material.

References

[AMO93] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[Ber98] D. P. Bertsekas, *Network Optimization: Continuous and Discrete Models*, Athena Scientific, Belmont, MA, 1998.

[BHT87] D. P. Bertsekas, P. A. Hosein, and P. Tseng, *Relaxation methods for network flow problems with convex arc costs*, SIAM Journal on Control and Optimization, 25 (1987), pp. 1219–1243.

[BeT89] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, London, U.K., 1989.

[Bru84] P. Brucker, *An $O(n)$ algorithm for quadratic knapsack problems*, Operations Research Letters, 3 (1984), pp. 163–166.

[CeZ97] Y. Censor and S. A. Zenios, *Parallel Optimization: Theory, Algorithms, and Applications*, Numerical Mathematics and Scientific Computing, Oxford University Press, Oxford, U.K., 1997.

[Cur93] N. D. Curet, *On the dual coordinate ascent approach for nonlinear networks*, Computers and Operations Research, 20 (1992), pp. 133–140.

[Eve63] H. Everett, III, *Generalized Lagrange multiplier method for solving problems of optimum allocation of resources*, Operations Research, 11 (1963), pp. 399–417.

[Fis81] M. L. Fisher. The Lagrangian relaxation method for solving integer programming problems. *Management Science*, 27 (1981), pp. 1–18.

[HKL80] R. V. Helgason, J. L. Kennington, and H. Lall, *A polynomially bounded algorithm for a singly constrained quadratic program*, Mathematical Programming, 18 (1980), pp. 338–343.

[IbK88] T. Ibaraki and N. Katoh, *Resource Allocation Problems: Algorithmic Approaches*, no. 4 in Foundations of Computing Series, The MIT Press, Cambridge, MA, 1988.

[Ius03] A. N. Iusem, *On the convergence properties of the projected gradient method for convex optimization*, Computational and Applied Mathematics, 22 (2003), pp. 37–52.

[KNS74] D. Klingman, A. Napier, and J. Stuts, *NETGEN: A program for generating large-scale capacitated assignment, transportation and minimum cost network flow problems*, Management Science, 20 (1974), pp. 814–821.

[LaL97] T. Larsson and Z. Liu, *A Lagrangean relaxation scheme for structured linear programs with application to multicommodity networks flows*, Optimization, 40 (1997), pp. 247–284.

[LaP06] T. Larsson and M. Patriksson, *Global optimality conditions for discrete and nonconvex optimization with applications to Lagrangian heuristics and column generation*, Operations Research, 54 (2006), pp. 436–453.

[LiP87] Y. Y. Lin and J.-S. Pang, *Iterative methods for large convex quadratic programs: A survey*, SIAM Journal on Control and Optimization, 25 (1987), pp. 383–411.

[LuT93] Z.-Q. Luo and P. Tseng, *On the convergence rate of dual ascent methods for linearly constrained convex minimization*, Mathematics of Operations Research, 18 (1993), pp. 846–867.

[Mar93] J. Marklund, *A study of Lagrangian heuristics for convex network flow problems*, Master's thesis, Department of Mathematics, Linköping Institute of Technology, Linköping, Sweden, 1993.

[Min60] G. J. Minty, *Monotone networks*, Proceedings of the Royal Society. London. Series A: Mathematical, Physical and Engineering Sciences, 257 (1960), pp. 194–212.

[OhK84] A. Ohuchi and I. Kaji, *Lagrangian dual coordinatewise maximization algorithm for network transportation problems with quadratic costs*, Networks, 14 (1984), pp. 515–530.

[Ols05] C. Olsson, *Lagrangian heuristics for strictly convex quadratic minimum cost network flow problems*, Master's thesis, Department of Mathematics, Chalmers University of Technology, Gothenburg, Sweden, Gothenburg, Sweden, 2005.

[Pat94] M. Patriksson, *The Traffic Assignment Problem—Models and Methods*, Topics in Transportation, VSP BV, Utrecht, The Netherlands, 1994. Available electronically at http://www.math.chalmers.se/˜mipat/traffic.html.

[Pat07] M. Patriksson, *A survey on the continuous nonlinear resource allocation problem*, European Journal of Operational Research, 2007, to appear.

[Roc70] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.

[Roc84] R. T. Rockafellar, *Network Flows and Monotropic Optimization*, John Wiley & Sons, New York, NY, 1984. Also published by Athena Scientific, Belmont, MA, 1998.

[Ven91] J. A. Ventura, *Computational development of a Lagrangian dual approach for quadratic networks*, Networks, 21 (1991), pp. 469–485.