Trade-off Curves for QoS Routing

Piet Van Mieghem
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
P.VanMieghem@ewi.tudelft.nl

Lieven Vandenberghe
Department of Electrical Engineering
University of California Los Angeles (UCLA)
vandenbe@ee.ucla.edu

***Abstract*— Trade-off curves for the exact and the relaxed QoS routing problem are presented and discussed. In addition, an efficient parametric linear programming algorithm to compute the trade-off curve of the relaxed problem is given. Trade-off curves enable a network operator to choose the set of constraints according to its QoS portfolio and are useful in the design of the network. In particular, the trade-off curves represent for the operator's network the possible basic feasible solutions that can be computed rapidly. In some sense, we reverse the QoS routing problem by giving the network operator the means to advertise an appropriate set of QoS constraints for its network. Instead of offering the user the freedom to require desirable end-to-end QoS levels from the operator, the user now can choose from the advertised QoS constraints portfolio those that best fit his application. The trade-off curve of the approximate, relaxed problem also gives insight in the computational complexity of QoS routing.**

I. Introduction

Consider a graph with N nodes and L links where each link l is specified by a link weight vector with m components $w_i(l)$ for $1 \leq i \leq m$. Assuming only additive[1] link weights, QoS routing determines the "best" path P from a source to a destination that satisfies

$$w_i(P) = \sum_{l \in P} w_i(l) \leq C_i \tag{1}$$

where C_i (with $1 \leq i \leq m$) are the end-to-end QoS constraints requested by the user. Before detailing what is meant by a "best" path, we write the QoS routing problem as an integer programming problem (see e.g. [1]). For clarity of the presentation, we confine the discussion to $m = 2$ link weights. We first introduce an integer flow vector x with L components and require that $x_l = 1$ if link l belongs to the path P, otherwise $x_l = 0$. Then, the QoS constraints (1) are compactly written as $w_1(P) = c^T x \leq C_1$ and $w_2(P) = d^T x \leq C_2$, where c and d are the vectors of all the first, respectively second, link weights in the graph. The QoS routing problem with $m = 2$ link weights is

$$\begin{array}{ll} \text{find} & x \\ \text{subject to} & Bx = b \text{ and } x_l \in \{0,1\} \text{ for all } 1 \leq l \leq L \\ & c^T x \leq C_1, d^T x \leq C_2 \end{array} \tag{2}$$

[1]The other type of links such as, for example, available capacity, are min(max) link weights as explained in [10] which are accommodated by topology filtering, i.e. all links that do not satisfy the min(max) QoS constraints are erased.

where B is the $N \times L$ node-link incidence matrix of the graph and b is the demand or supply vector. If $b_i = 1$, $b_j = -1$ and $b_k = 0$ for $k \neq i, j$, then the solution of (2) is a feasible QoS path P from node i to node j. If more feasible paths exist, an additional criterion can distinguish the "best" path.

Problem (2) with two or more additive link weights ($m \geq 2$) is proved to be NP-complete by Wang and Crowcroft [12]. However, already in 1979, Hansen [4] has proved this fact. Moreover, Hansen's arguments show that, although in QoS routing the constraints C_1 and C_2 play a crucial role as opposed to multiple criteria path problems, the same essential difficulties leading to NP-completeness appear in both types of problems.

II. The trade-off curve

Instead of (2), we consider the multi-objective integer programming problem

$$\begin{array}{ll} \text{minimize} & \left(c^T x, d^T x\right) \\ \text{subject to} & Bx = b \text{ and } x_l \in \{0,1\} \text{ for all } 1 \leq l \leq L \end{array} \tag{3}$$

where the QoS constraints (1) are omitted. We will return later to the QoS constraints C_1 and C_2.

A feasible solution x of (3) satisfies $Bx = b$ and $x_l \in \{0,1\}$ for all $1 \leq l \leq L$ and optimal solutions are those that in addition minimize the vector objective function whose meaning we now explain. Figure 1 shows in dark shaded circles the $w_1 = c^T x$ and $w_2 = d^T x$ values of the feasible set of solutions of (3) in the w_2, w_1-plane. The light shaded circles are Pareto optimal values of the Pareto optimal flow vectors or the non-dominated achievable QoS specifications[2]: each of these Pareto optimal values minimizes (w_1, w_2) and none of them is dominated by another feasible value. The staircase curve that connects these Pareto optimal points is called the trade-off curve. The achievable QoS specifications can lie on the trade-off curve: if they lie on a line segment, they are dominated by one Pareto optimal value, while on a corner (as in Figure 1), they are dominated by precisely two (consecutive) Pareto optimal values. The area above the trade-off curve is the region of achievable QoS specifications and a point below the trade-off curve would represent an unachievable QoS specification.

[2]A vector $y = (y_1, \ldots, y_m)$ is dominated by a vector $x = (x_1, \ldots, x_m)$ if $x_j \leq y_j$ for all $1 \leq j \leq m$ and where there is at least one strict inequality.

1-4244-0222-0/06/$20.00 (c)2006 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the Proceedings IEEE Infocom.

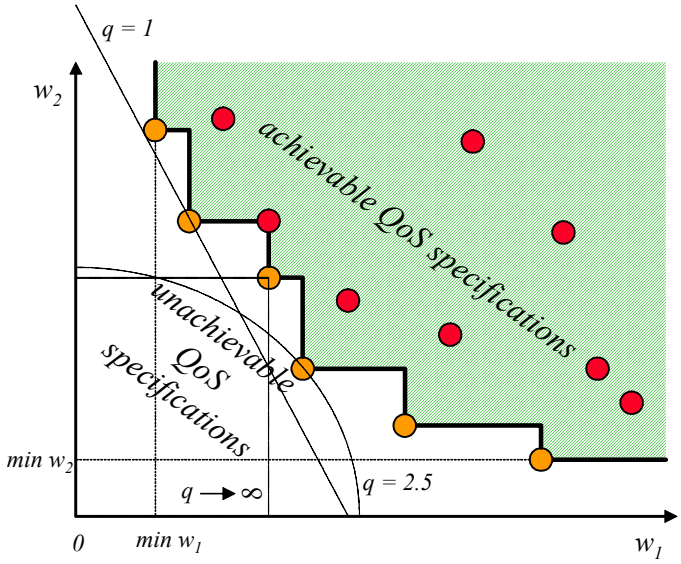

Fig. 1. The mapping of the Pareto optimal flow vectors x into the w_2, w_1-plane (light shaded circle). The staircase line connecting these Pareto optimal values is the trade-off curve. The achievable QoS specifications (dark shaded circles) lie in the region above the trade-off curve.

In order to find Pareto optimal points, scalarization (see e.g. [3, Sec. 4.7.4]) is commonly used. Problem (3) minimizes a vector. Scalarization maps the vector to a real number by introducing a minimization criterion. In $m = 2$ dimensions, the map is $\mathbb{R}^2 \to \mathbb{R} : \left(c^T x, d^T x\right) \to f\left(c^T x, d^T x\right)$ where f obeys certain monotonicity properties [3, Sec. 4.7]. The scalar optimization problem

$$\begin{aligned} &\text{minimize} \quad f\left(c^T x, d^T x\right) \\ &\text{subject to } Bx = b \text{ and } x_l \in \{0,1\} \text{ for all } 1 \le l \le L \end{aligned} \tag{4}$$

returns an optimal solution $x^*(f)$. In the design of the exact QoS routing algorithm SAMCRA [10], the scalarization f was the weighted Hölder q-norm, written in $m = 2$ dimensions as

$$f\left(\left(c^T x, d^T x\right), q\right) = \left(\left(\frac{c^T x}{C_i}\right)^q + \left(\frac{d^T x}{C_2}\right)^q\right)^{1/q} \tag{5}$$

Figure 1 shows schematically for a few values of the free parameter q (and some values of C_1 and C_2 in (5)) which Pareto optimal values correspond to optimal solutions of (4) specified by (5). Scalarization thus allows us to incorporate the QoS constraints as in SAMCRA. Instead of considering all possible values of q, in SAMCRA, only the case $q \to \infty$ is computed as motivated in [10].

III. Bounds

A. Relaxation

An obvious scalarization function f is a linear function in the vector components $f = \lambda_1 c^T x + \lambda_2 d^T x$ where λ_1 and λ_2 are positive real numbers that will be specified below. A fundamental result in network optimization (see e.g. [1]) states that the relaxation of $x_l \in \{0,1\}$ to $0 \le x_l \le 1$ in the problem (4) still leads to exact results. By omitting the requirement that the flow vector should have integer components (needed to represent a path), the relaxed vector is still exact. Hence, the problem (4) is reduced to

$$\begin{aligned} &\text{minimize} \quad \lambda_1 c^T x + \lambda_2 d^T x \\ &\text{subject to } Bx = b \text{ and } 0 \le x_l \le 1 \text{ for all } 1 \le l \le L \end{aligned} \tag{6}$$

which will be simplified to the single parameter objective function $c^T x + \lambda d^T x$ because minimizing $\lambda_1 c^T x + \lambda_2 d^T x = \lambda_1 \left(c^T x + \frac{\lambda_2}{\lambda_1} d^T x\right)$ is identical to minimizing $c^T x + \lambda d^T x$ where $\lambda = \frac{\lambda_2}{\lambda_1}$ since both $\lambda_{1,2} > 0$. Problem (6) has an efficient solution as shown in the Appendix.

By varying $\lambda \ge 0$, we thus find Pareto optimal values of the original problem (3). However, even by varying λ over all possible non-zero numbers, not all Pareto optimal values of (3) are guaranteed to be found. Since here lies a possible approximation, we explain this point in more detail. The feasible set of vectors x that satisfy[3] $Bx = b$ and $0 \preceq x \preceq 1$ in (6) forms a polyhedron $\mathcal{P}$. The linear map $f = c^T x + \lambda d^T x$ projects the edge vectors x of the polyhedron $\mathcal{P}$ onto the vector $c + \lambda d$ and the minimum of those projections for each λ is maintained. We can plot the minimum of the objective function $c^T x + \lambda d^T x$ in a $w_1 = c^T x$ and $w_2 = d^T x$ plane as a function of λ.

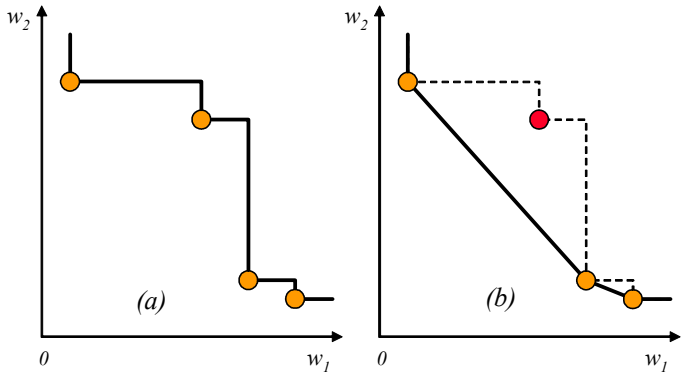

Fig. 2. The trade-off curve of the exact problem (a) and of the relaxed problem (b). In (b), the convex hull of the achievable QoS specification is found.

In summary, the trade-off curve of the relaxed problem (6) is a piecewise-linear curve $g(\lambda)$ because it is the lower part of the boundary of the image of the polyhedron $\mathcal{P}$ under the linear mapping $\left(c^T x, d^T x\right)$. Figure 2 exemplifies a situation where only three of the in total four Pareto optimal values are found, due to the non-convex nature of the original integer programming problem.

B. The trade-off curve of the relaxed problem

In general for any $m > 1$, the objective function is $\lambda^T w$ where λ is an m-dimensional vector as well as w. If $\lambda^T w = a$

[3]Following the notation in [3], recall that $x \succeq 0$ means that each vector element in x is nonnegative.

where a is a constant, it shows that the vector λ is orthogonal to the vector w. A schematic of the piecewise linear trade-off curve is shown in Figure 3. If $\lambda_1 = 0$ (or $\lambda \to \infty$), we find the shortest path in a graph with a single link weight given by the second component of the link weight vector $\vec{w}$. Hence, in Figure 3, the line parallel to the w_1-axis gives the weight of that shortest path and, similarly, the line parallel to the w_2-axis corresponds to the weight of the shortest path in the first component of the link weight vector w only ($\lambda = 0$). These values (min w_1, min w_2) are the absolute minimum for (w_1, w_2) and readily obtained by e.g. Dijkstra's shortest path algorithm.

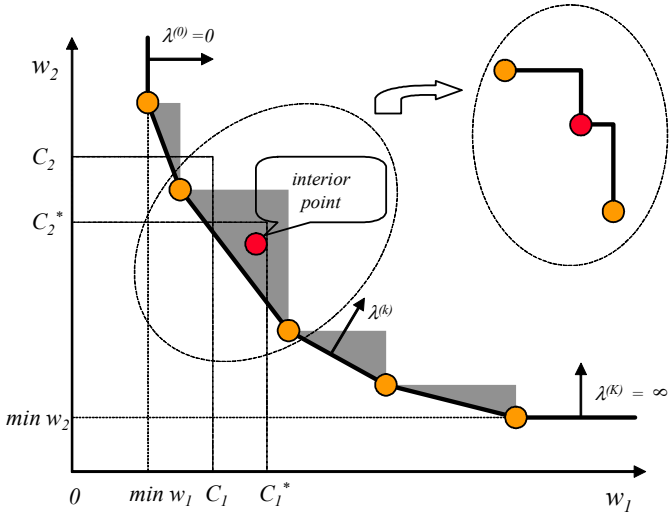

Fig. 3. The feasible region lies above the trade-off curve (bold line) for the Problem 6. If only one vertex is enclosed by the constraint C_1 and C_2, it is the only solution with those QoS constraints. If the constraints C_1^* and C_2^* do not enclose an extreme point, the remaining triangular area needs to be scanned for a possible integer solution (i.e. a path). If there is only one interior extreme point between two consecutive extreme points on the trade-off curve, the ellipsoidal insert shows the resulting exact trade-off curve between these extreme points.

Since $g(\lambda) = \min_x c^T x + \lambda d^T x$ is the optimal value of the relaxation, the QoS routing problem that additionally imposes a QoS constraints vector C is lower bounded by the trade-off curve $g(\lambda)$ for $\lambda \geq 0$ of the relaxed problem.

A different formulation of (6) is

$$\begin{aligned} &\text{minimize} \quad d^T x \\ &\text{subject to } Bx = b \text{ and } 0 \preceq x \preceq 1 \text{ and } c^T x \leq \gamma \end{aligned} \tag{7}$$

where the optimum $x^*(\gamma)$ will be a function of the QoS constraint γ rather than the parameter λ that does not have an immediate interpretation. The representation of $w_2 = d^T x^*(\gamma)$ versus γ will be exactly the same as the piece-wise linear curve shown in Figure 3 but with a different parametrization (γ instead of λ). Thus, $\gamma \to \infty$ corresponds to $\min w_2$ and $\gamma = \min w_1$ corresponds to the point $(\min w_1, w_2)$. Values of $\gamma < \min w_1$ will lead to infeasible solutions.

C. Upper and lower bounds

The piece-wise linear nature of $g(\lambda)$ means that a basic feasible solution x_j^* remains the minimum in the λ-interval $[\lambda_j, \lambda_{j+1})$. Applied to the shortest path problem, a certain path P_j (represented via the flow x_j^*) is the shortest one during a change of the link weight vector from $(c + \lambda_j d)$ to $(c + \lambda_{j+1} d)$ with $\lambda_{j+1} > \lambda_j$ for all $1 \leq j \leq K$, where K is the number of extreme points (or basic feasible solutions). From a practical viewpoint, the range $[\lambda_j, \lambda_{j+1})$ reflects the stability regime in which the path P_j is the shortest and the increasing term $\lambda d^T x$ can be considered as an external influence on the link weights due to, e.g. changing traffic conditions that enlarge the delay, loss etc., or due to the changing of a pricing scheme. If all K extreme points are known with their corresponding active λ-region, topology update rules for recomputing the shortest path may be triggered to start flooding when a minimum change $\Delta\lambda_{\min} = \min_j (\lambda_{j+1} - \lambda_j)$ is exceeded. However, as will be shown below, there are cases in which $\Delta\lambda_{\min} = 0$, and, in general, as N grows large, $\Delta\lambda_{\min} \to 0$. This means that in large networks shortest paths are intrinsically unstable.

Given a topology via the incidence matrix B, it is, in general, as explained in [8], a difficult problem to estimate analytically the number K of extreme points. If the number of extreme points is large, the piecewise-linear trade-off curve $g(\lambda)$ tends to a smooth differentiable (and convex) curve which can be fitted to provide a compact analytical form for the lower bound of the QoS routing problem (2). If only a few extreme points are known, the piecewise-linear nature of the trade-off curve of the relaxed problem allows us to bound the region of feasible solutions by linear segments from the $\lambda = 0$ and $\lambda \to \infty$ extreme points as illustrated in Figure 4. Moreover, since the left and right derivative of $g(\lambda)$ at a basic feasible solution x_j^* are λ_j and $\lambda_{j+1} > \lambda_j$, respectively, the tangent at x_j^* computed as the mean of the left and the right derivative bounds the feasible regions of the relaxed problem from below, whereas the linear segment through the $\lambda = 0$ and $\lambda \to \infty$ extreme points bounds that feasible region from above.

Figure 4 also shows that there are two extreme types of trade-off curves. The first one consists of a right angle corner connecting the $\lambda = 0$ and $\lambda \to \infty$ extreme points. The intersection of the horizontal line and vertical line through these two extreme points is the extreme point with coordinates $(\min w_1, \min w_2)$. In this extreme case, that latter point is precisely the only optimal point and in fact there is nothing to trade-off. The second extreme case consists of the line connecting the $\lambda = 0$ and $\lambda \to \infty$ extreme points: all extreme points on that line are Pareto-optimal: if the w_1 value of an extreme point is decreased, the w_2 value is necessarily increased. Clearly, all other trade-off curves of the relaxed problem lie in between these extremes.

The trade-off curve of the relaxed problem also contains information to upper bound the exact trade-off curve of the original problem. The dark shaded triangles above the bold

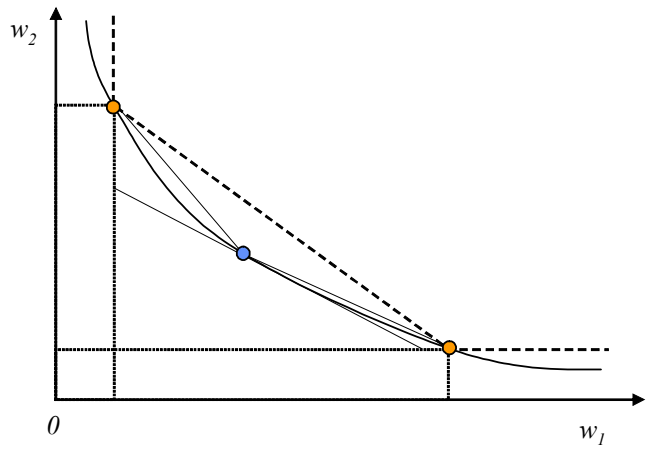

Fig. 4. The continuous, convex function $w_2 = g_c(w_1)$ passing through the $\lambda = 0$ and $\lambda \to \infty$ point. The tangent to an arbitrary point on g_c reduces the search space in the (w_1, w_2) plane of feasible solutions to all points above the tangent.

trade-off curve in Figure 3 represent the largest possible regions in which an optimal interior point of problem (3) can lie. Any other interior point is not optimal because it is dominated by an extreme point on the trade-off curve. The upper bound curve to the optimal solution is a staircase curve (Figure 1) obtained by connecting the extreme points via the right angle edges of each triangle instead of by the triangle's hypothenusa (which gives the trade-off curve of the relaxed problem). Hence, the area of the triangle and related to this, the distance between two consecutive extreme points on the trade-off curve, characterize the maximum degree by which w_1 and w_2 of a possible optimal interior solution can deviate from that of neighboring extreme points. If the area is small, the extreme points on the trade-off curve of the relaxed problem are very close to the optimal solution of (3). If we know an interior optimal point as illustrated in Figure 3, all possible other solutions in the rectangular area of the triangle above and to the right of the interior optimal point are dominated by that interior point. Hence, the staircase curve should be reduced by a rectangular inlet at interior optimal points. If there is only one interior point between two extreme points on the trade-off curve, the optimal curve is shown in the ellipsoidal insert in Figure 3. Compared to the exact trade-off curve shown in Figure 1, the line connecting both extreme points on the trade-off curve of the relaxed problem is thus clearly a lower bound (and can be erased).

D. The trade-off curve of the relaxed problem and QoS constraints

As an interesting application of the relatively easily computed trade-off curve of the relaxed problem, a network provider may offer to customers in his QoS portfolio only those constraints for which one extreme point lies within the QoS (C_1, C_2) constraints.

When given the freedom to choose (C_1, C_2), there are constraints for example (C_1^*, C_2^*) in Figure 3 that do not enclose an extreme point. This means that, for such constraints, the trade-off curve does not provide a solution for a path or spanning tree and the optimal path/tree is an interior point solution of (3) inside the polyhedron $\mathcal{P}$ of the relaxed problem. In general there are no efficient search methods to find such an optimal interior point. This observation also explains why there are possible cases in which the QoS routing problem is difficult, i.e. NP-complete. The points on the line between two extreme points which lie within the (C_1^*, C_2^*) are feasible flows, but not paths. Assuming that we have a train of packets to send from a source to a destination, we may interpret such solution as a flow over several paths and the x_j component gives the fraction of packets that travel over link j. If a packet train can be efficiently routed over several paths in the network, this solution is optimal.

IV. Computation of the trade-off curve of the relaxed problem

A general parametric linear programming algorithm (PLPA) to compute the trade-off curve of the relaxed problem is presented and explained in the Appendix A. In the sequel, only the trade-off curve of the relaxed problem (6) is considered further. Here we limit ourselves to explaining the PLPA by an example in Section IV-A.

In order to avoid degenerate extreme point solutions (explained in the Appendix A), we first compute the shortest path tree rooted at the source node (with label 1) to all other nodes in the network for which the demand vector is $b^T = (N-1, -1, -1, \ldots, -1)$. From this shortest path tree, all possible shortest paths from that source to any other node in the network are readily obtained.

A. Example

In this section, we illustrate the PLPA by a simple network example. We use the convention that a flow at a node j is positive when unit traffic is injected at node j into the network and negative if unit traffic leaves the network at node j. Information about the direction of the links is specified by the *incidence matrix* B, a $N \times L$ matrix with elements

$$b_{ij} = \begin{cases} 1 \text{ if link } e_j = i \longrightarrow j \\ -1 \text{ if link } e_j = i \longleftarrow j \\ 0 \text{ otherwise} \end{cases}$$

The potential y, defined in the Appendix A, is negative in each point, except for the source in which current is injected in the network. This convention is similar as in electrical networks.

A simple network is shown in Figure 5.

The numbering of the links is given in Figure 6. The corresponding incidence matrix $B_{5\times 10}$ is

$$B = \begin{bmatrix} -1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 \\ 0 & 0 & -1 & 0 & -1 & 1 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & -1 \end{bmatrix}$$

and both link weight vectors c and d are

$$c^T = \begin{bmatrix} 3.1 & 2.3 & 1.2 & 12.3 & 2.5 & 9.6 & 2.1 & 6.8 & 2.1 & 2.2 \end{bmatrix}$$
$$d^T = \begin{bmatrix} 4.2 & 7.4 & 14.7 & 9.7 & 1.0 & 3.1 & 5.1 & 2.2 & 5.9 & 6.7 \end{bmatrix}$$

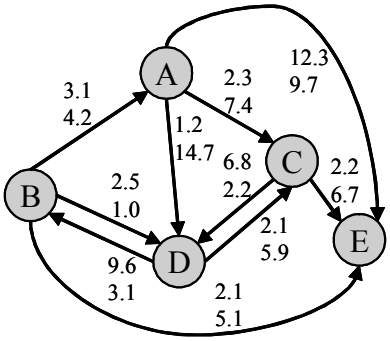

Fig. 5. Example network.

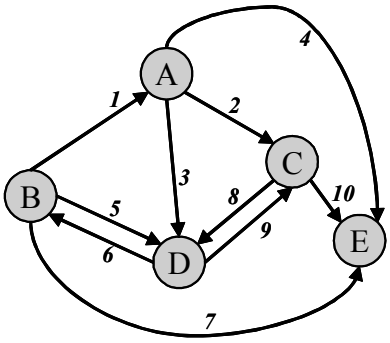

Fig. 6. The numbering of the links in the example network.

The initial case for $\lambda^{(0)} = 0$ is shown in Figure 7. The

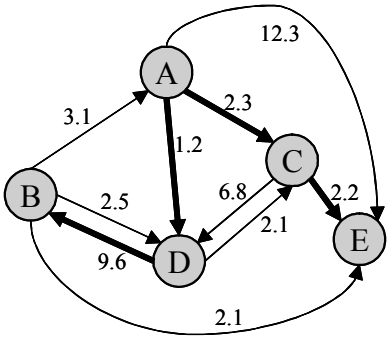

Fig. 7. Network weights with $\lambda^{(0)} = 0$.

corresponding potential vector is

$$\left(y^{(0)}\right)^T = \begin{bmatrix} 0 & -10.8 & -2.3 & -1.2 & -4.5 \end{bmatrix}$$

and the corresponding flow vector is

$$\left(x^{(0)}\right)^T = \begin{bmatrix} 0 & 2 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

If each node in the network is once a source, then summing all flow vectors results in an aggregate flow vector of which each non-zero flow component gives the betweenness of that link, defined as the number of shortest paths each with unit traffic that traverse a link

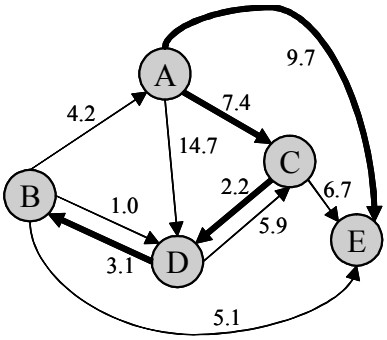

Fig. 8. Network weights with $\lambda^{(K)} \to \infty$.

The final case for $\lambda^{(K)}$ is shown in Figure 8. The corresponding potential vector is

$$\left(y^{(K)}\right)^T = \begin{bmatrix} 0 & -12.7 & -7.4 & -9.6 & -9.7 \end{bmatrix}$$

and the corresponding flow vector is

$$\left(x^{(K)}\right)^T = \begin{bmatrix} 0 & 3 & 0 & 1 & 0 & 1 & 0 & 2 & 0 & 0 \end{bmatrix}$$

Starting from $x^{(0)}$, the set I which consists of rows of the non-zero elements in $x^{(0)}$ equals $I = \{2, 3, 6, 10\}$. We find $\lambda^+ = \lambda^{(1)} = 1.549$ and the corresponding flow vector is

$$\left(x^{(1)}\right)^T = \begin{bmatrix} 0 & 4 & 0 & 0 & 0 & 1 & 0 & 2 & 0 & 1 \end{bmatrix}$$

This network situation is shown in Figure 9. The corresponding potential vector is found as a solution of $y(\lambda) = B_I^{-T}(c_I + \lambda d_I)$, which is explicitly,

$$\left(y^{(1)}\right)^T = \begin{bmatrix} 0 & -38.3725 & -13.7627 & -23.9706 & -26.3412 \end{bmatrix}$$

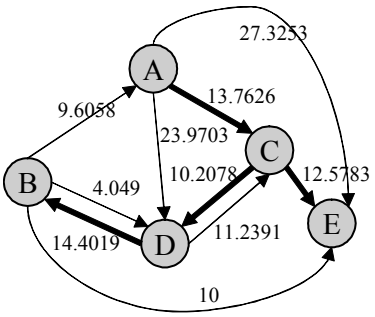

Fig. 9. The network weights if $\lambda = \lambda^{(1)} = 1.459$.

In the next step $k = 2$, we find $\lambda^+ = \lambda^{(2)} = 1.7727$ with corresponding flow vector $x^{(2)} = x^{(K)}$. Hence, in this example, there are only three extreme points in the $d^T x$ versus $c^T x$ plot as illustrated in Figure 10. The difference between the slopes $\lambda^{(2)} - \lambda^{(1)} \approx 0.23$ is too small to be visible in Figure 10.

The algorithm gives the total flow information from the source to each node. By separating the source tree into

the individual shortest paths from source to each node, the trade-off curve for each shortest path from a same node is determined and shown in Figure 10. In this example, we see that the constraints $C_1 = 20$ and $C_2 = 20$ are sufficient to allow that each extreme point lies within these constraints. Hence, for a multicast session with the source at node 1, the constraints $(20, 20)$ and, even $(12, 20)$ for $\lambda = 0$ only, can be guaranteed. From the flow vector, we can compute

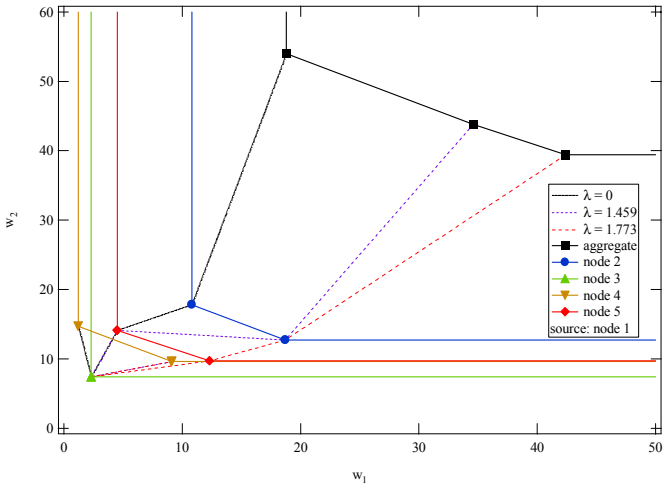

Fig. 10. The trade-off curves for the example network. The aggregate is the sum of the trade-off curves of the individual nodes.

the *betweenness*. The betweenness gives indications about the importance or centrality of a link. Load balancing would strive to equalize the betweenness over the links in the network as much as possible. Hence, instead of choosing the shortest path tree specified by λ that satisfies certain QoS constraints, we may choose that $\lambda = \lambda^{(k)}$ with the most uniform flow vector $x^{(k)}$ which corresponds to $\lambda = 0$ in the example network.

B. The number K of basic feasible solutions

The PLPA does not guarantee that all possible basic feasible solutions $x^{(k)}$ belonging to $\lambda = \lambda^{(k)}$ are found. We have already explained that the trade-off curve of the relaxed problem is the image of a linear mapping of the feasible polyhedron $\mathcal{P}$ on the w_1,w_2-plane. If that polyhedron $\mathcal{P}$ has facets parallel to $c + \lambda d$, all basic feasible solutions (vertices of the feasible polyhedron $\mathcal{P}$) are projected onto a same line segment in the w_1,w_2-plane. At a vertex in such a facet, there are necessarily at least two edges connecting that vertex to others in that facet, but more are possible. In the latter case, the PLPA algorithm chooses only one of the possible bifurcational arcs and continues determining all vertices (extreme points) on that arc, but it neglects all possible others on the second (or other) arc(s). Therefore, the PLPA may be used as an approximate and indicative tool, not to solve non-linear problems exactly. For example, the PLPA does not necessarily enumerate all possible shortest hop trees rooted at some node.

V. Analysis of two network types

Recently, Kuipers and Van Mieghem [6] have determined conditions on the graph G and the link weight structure that make the QoS routing problem (2) hard. From a practical point of view, if we can find the set of all networks for which the QoS routing problem is hard, we can propose guidelines to choose the link weight structure (or the graph if possible) such that QoS routing is tractable. The latter means that a QoS multi-constrained path $P_{A \to B}$ can be computed exactly, e.g. with SAMCRA [10], with a complexity that increases polynomially with the size of the graph (i.e. with N and/or L). In fact, as shown in [5], many instances of the QoS routing problem only require a computational complexity that is slightly larger than Dijkstra's shortest path algorithm. In this section, we present observations deduced from the trade-off curve analysis for two different classes of graphs: the small-world Erdös and Rényi random graph $G_p(N)$ [2] which was found to be "easy" and the lattice which is a large diameter graph and was found to be "difficult".

If we create a random matrix $A_{N \times L}$ (which is not necessarily equal to an incidence matrix B that represents a network) and uniformly distributed link weight vectors c and d, then we find for sufficiently large N, that the trade-off curve tends to a smooth differentiable curve. This means that $\lambda_{\min} = \min_{1 \le k \le K} \lambda^{(k)}$ tends to zero. In particular, for the random graph $G_p(N)$, we find that the values $\lambda^{(k)}$ are roughly exponentially increasing in k and that the total number of extreme points K is increasing in the number of nodes N approximately as a power law. For the lattice, we also found that $K \approx cN^a$ (where $a, c > 0$). Figure 11 shows a typical

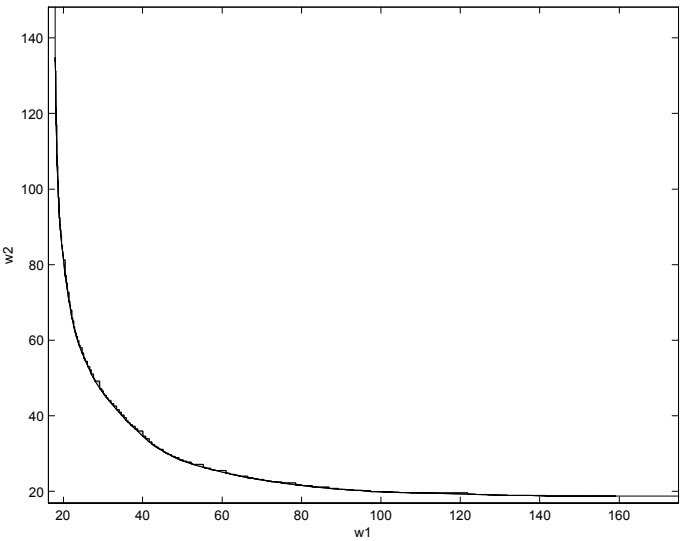

Fig. 11. A typical trade-off curve for $G_{0.3}(64)$ where both link weight vectors c and d are not correlated and uniformly distributed on [0,1].

trade-off curve for the total shortest path tree rooted at the source in the random graph $G_p(N)$ where $p = 0.3$ and $N = 64$. Figure 12 shows the related trade-off curve of the shortest path from the source to an arbitrary node (here node

64) in the same random graph as that in Figure 11. The main difference is the large number K of extreme points in the total shortest path tree (which results in an almost everywhere differentiable curve) as opposed to the relatively small number of extreme points for the case of a single path.

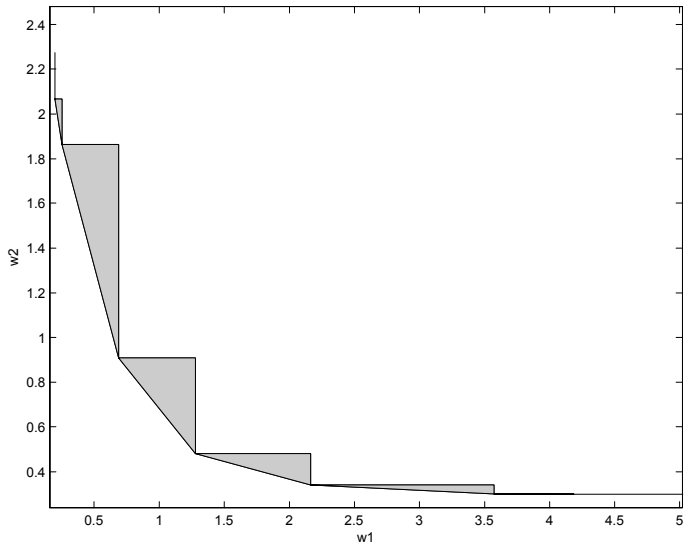

Fig. 12. The trade-off curve of the shortest path from the source node to node 64 in the same graph as Figure 11.

Instead of uncorrelated link weight vectors c and d, in the sequel, we consider perfectly negatively correlated uniformly distributed link weight vectors such that $d = 1 - c$ because negative correlation was found in [6] as a condition that makes the QoS problem hard.

A. Lattices

If we create a lattice with size z_1 and z_2 containing $N = (z_1 + 1)(z_2 + 1)$ nodes where c is uniformly distributed on [0,1] and $d = 1 - c$, we observe an almost "three-lines" piecewise plot as illustrated in Figure 13 for the shortest path tree and in Figure 14 which shows the corresponding path from the source to an arbitrary node.

The diagonal line is characterized by a single λ, which means that $(c + \lambda(1 - c))^T x^{(k)}$ is constant for all these points, but that $w_2 = d^T x^{(k)}$ and $w_1 = c^T x^{(k)}$ are different. This single λ equals 1, such that $(c + \lambda d)^T x^{(k)} = 1^T x^{(k)} = \left\| x^{(k)} \right\|_1$ and the optimization criterion is independent of the link weights c and, thus a fundamental property of a lattice. If we fix the underlying topology – N is fixed in the case of a lattice –, all M_t different, optimal flow vectors from a source node to all other nodes that minimize the sum of the components $1^T x^{(k)}$, correspond to all M_t possible minimum hop trees rooted at the source. Since all components of $x^{(k)}$ are integers, $1^T x^{(k)}$ is the same integer for all these flow vectors. In case we confine to paths (instead of trees), the number of minimum hop paths between corner points on a diagonal in the lattice is $M_p = \binom{z_1+z_2}{z_1}$ and $1^T x = z_1 + z_2$ (see e.g. [9]). In any lattice with N nodes and perfectly negatively correlated

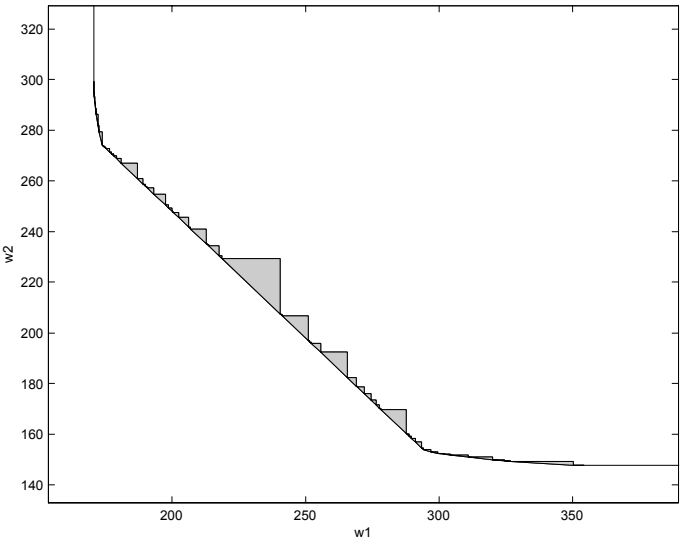

Fig. 13. A typical trade-off curve for a lattice with size $z_1 = z_2 = 7$ (or $N = 64$) and where the link weights are uniformly distributed but perfectly negatively correlated.

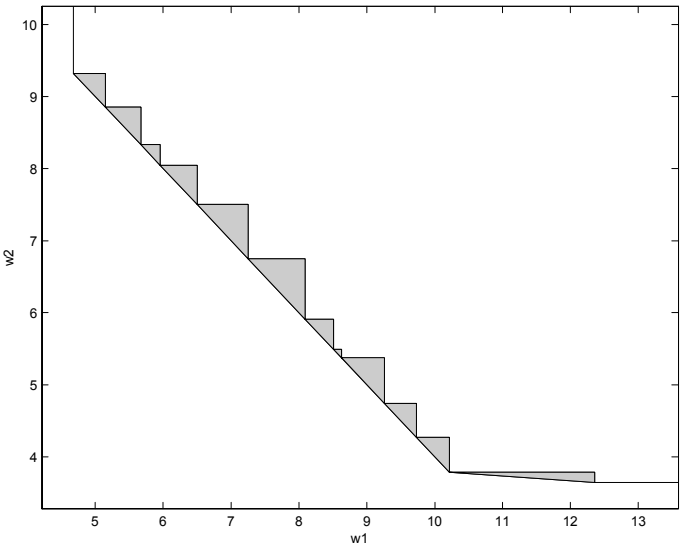

Fig. 14. The trade-off curve of a path in the same lattice as that of Figure 13

link weights ($d = 1 - c$), the exact trade-off curves consists of precisely M_p points on the $\lambda = 1$ line. This constant number M_p is not returned in the trade-off curves computed by PLPA due to the choice of PLPA at bifurcational arcs as explained earlier. If it were, the line segment corresponding to $\lambda = 1$ would be a uniform weighted $x^{(k)}$ of precisely a same set of M_p flow vectors. Hence the spacing should be almost exponential with mean $\frac{1}{M_p}$ by a same argument as used in [11, Section II.c]. The number M_p of these flow vectors is not constant in each instance and the spacing distribution is not entirely exponential as observed in Figure 15 and 16.

Figure 13 (and 14) clearly shows the grey triangles that

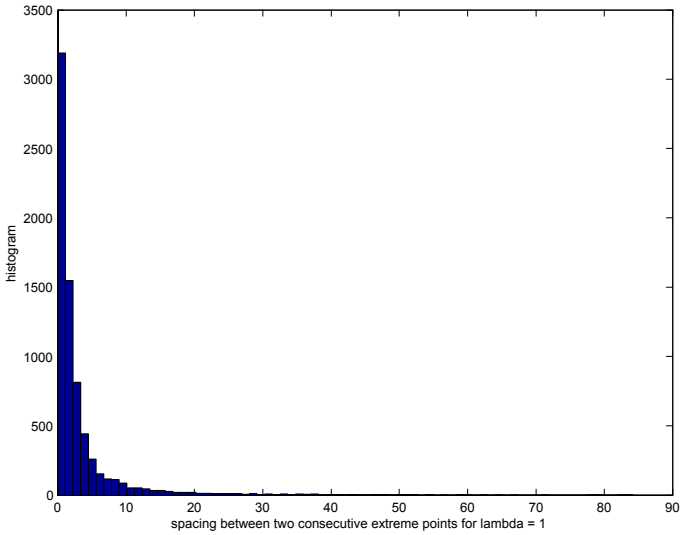

Fig. 15. The histogram of the spacing between two consecutive extreme points on the line segment of the trade-off curve corresponding to $\lambda = 1$ for the shortest path tree in the lattice with $N = 64$. The mean is 3.23 and the standard deviation is 6.18. The number of instances simulated is 100.

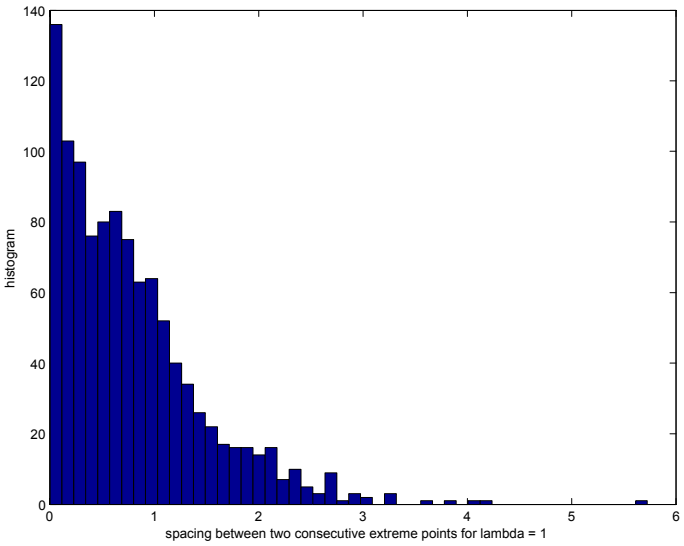

Fig. 16. The histogram of the spacing between two consecutive extreme points on the line segment of the trade-off curve corresponding to $\lambda = 1$ for a single path in the lattice with $N = 64$. The mean is 0.79 and the standard deviation is 0.69. The number of lattice instances simulated is 100.

characterize potential regions with interior point solutions of the original problem (3). If the QoS constraints are placed in such a segment, interior points need to be found which is generally computational intensive and which may explain NP-hard features. "Difficult" constraints (C_1^*, C_2^*) as illustrated in Figure 3 are lying between extreme points, i.e. $w_{1,j+1} \leq C_1^* \leq w_{1j}$ and $w_{2,j} \leq C_2^* \leq w_{2,j+1}$ where $w_{1,j} = c^T x_j^*$ and $w_{1,j} = d^T x_j^*$. In fact, a solution of problem (3) requires that we find integer solutions (that represent a path) of the flow vector x of which both $c^T x$ and $d^T x$ lie within that triangle. Intuitively, one would argue that the larger the triangle, the larger the expected search space and the larger the computation time. While probably correct in many network instances, unfortunately, the projective nature of the trade-off curve does not allow us to state this argument as a truth. We cannot exclude that extreme points of some polyhedron along some direction are all projected into a small triangle of the (w_1, w_2)-plane.

The exact trade-off curve corresponding to that in Figure 14 has been computed with SAMCRA. On the $\lambda = 1$ line, precisely $M_p = \binom{14}{7} = 3432$ extreme points appear (in contrast to $K - 1 = 12$ in Figure 14 computed by PLPA). The large number in extreme points (Pareto optimal points), $M_p = \binom{z_1+z_2}{z_1}$, illustrates the NP-hardness. On the $\lambda = 1$ segment, no interior points (in a triangle) are found. For perfectly negatively correlated link weights, an interior point must possess a hopcount larger than the minimum possible that is $h = z_1 + z_2$. For example, the second shortest hop path P_{h+2} between opposite corner nodes (consisting of $h+2$ hops) lies on the line $w_2(P_{h+2}) = h + 2 - w_1(P_{h+2})$ parallel to the $\lambda = 1$ line that obeys $w_2(P_h) = h - w_1(P_h)$. The probability that a point $(w_1(P_{h+2}), w_2(P_{h+2}))$ is not dominated by a point $(w_1(P_h), w_2(P_h))$ on the $\lambda = 1$ line seems very small for uniform link weights.

B. Erdös and Rényi random graph

A typical trade-off curve for $G_{0.3}(64)$ is shown in Figure 17 for the shortest path tree and in Figure 18 for the corresponding shortest path. In contrast to the lattice with the same number of nodes $N = 64$, for the random graph (which has about 5 times more links), the trade-off curve is almost differentiable and extreme points are more or less homogeneously distributed over the trade-off curve. The upper (staircase) and lower bound (trade-off curve) approach each other which means that the exact solution of the original QoS routing is very well approximated. However, also here, there is a diagonal line segment corresponding to $\lambda = 1$ which contains many extreme points, but this line segment is much less dominant compared to the lattice. An explanation is that, in $G_p(N)$, the minimum hop path does not have a fixed length as opposed to $z_1 + z_2$ in a lattice. The shorter hop path (between two nodes) dominates all other on the $\lambda = 1$ segment. Hence, the dominance principle efficiently reduces the possible number of feasible solutions. This may explain that, for all constraints, SAMCRA readily finds a solution and that NP-hard behavior is not observed in $G_p(N)$. The corresponding trade-off curve for a path has a quite high curvature, which classifies the trade-off curve close to the extreme single corner trade-off curve with a single optimum. Again this points to computational easy cases of the QoS routing problem. In contrast, the trade-off curves in Figure 13 and 14 for the lattice are near to the other extreme type of a trade-off curve.

VI. Summary

We have shown that the trade-off curve of the relaxed problem is useful to both upper and lower bound the exact

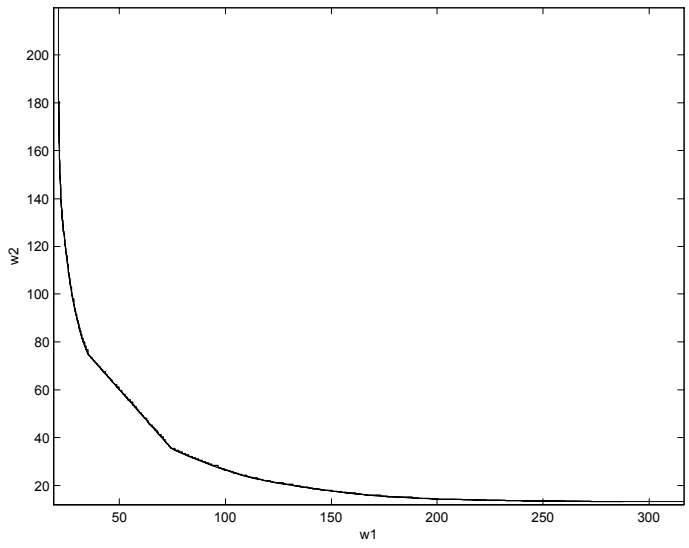

Fig. 17. A typcial trade-off curve for $G_{0.3}\,(64)$ with uniform link weights, but perfectly negatively correlated.

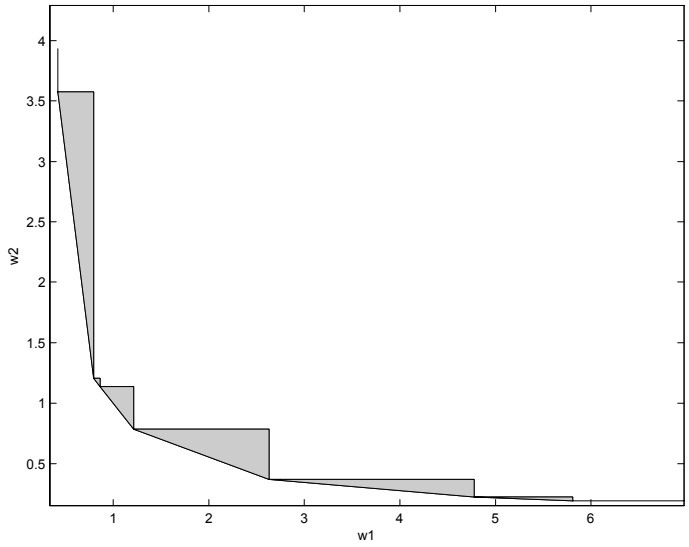

Fig. 18. A typcial trade-off curve of a path in the same graph $G_{0.3}\,(64)$ as that of Figure 17

solution of the QoS routing problem. In addition, the trade-off curve of the relaxed problem is efficiently computed by a parametric linear programming algorithm (PLPA). The Pareto values returned by the PLPA are also exact Pareto values of the original QoS routing problem. Beside choosing the set of QoS constraints, a network operator may also determine the operational point (specific λ) that leads to the most load-balanced situation. Finally, the amount of large triangles above and the overall curvature of the trade-off curve of the relaxed problem give indications about the possible computational difficulty of the QoS routing problem.

Acknowledgements. This work performed at UCLA was supported by the Dutch National Foundation for Applied Sciences, STW project DTC.6758. We would like to thank F. Kuipers for computing the exact trade-off curves with SAMCRA.

REFERENCES

[1] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, Belmont, MA, 1997.
[2] B. Bollobas. *Random Graphs*. Cambridge University Press, Cambridge, UK, 2nd edition, 2001.
[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2004.
[4] P. Hansen. Bicriterion path problems. *Multiple Criteria Decision Making, Theory and Application, ed. G. Fandel and T. Gal, Lecture Notes in Economics and Mathematical Systems 177, Springer-Verlag, Berlin*, pages 109–127, 1980.
[5] F. A. Kuipers, T. Korkmaz, M. Krunz, and P. Van Mieghem. Performance evaluation of constraint-based path selection algorithms. *IEEE Network*, 18(5):16–22, September/October 2004.
[6] F. A. Kuipers and P. Van Mieghem. Conditions that impact the Complexity of QoS Routing. *IEEE/ACM Transaction on Networking*, 13(4):717–730, August 2005.
[7] C. Roos, T. Terlaky, and J. P. Vial. *Theory and Algorithms for Linear Optimization: An Interior Point Approach*. John Wiley & Sons, 1997.
[8] A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, New York, 1986.
[9] P. Van Mieghem. *Performance Analysis of Communications Systems and Networks*. Cambridge University Press, 2006.
[10] P. Van Mieghem and F. A. Kuipers. Concepts of exact quality of service algorithms. *IEEE/ACM Transaction on Networking*, 12(5):851–864, October 2004.
[11] P. Van Mieghem and S. van Langen. Influence of the link weight structure on the shortest path. *Physical Review E*, 71:056113, May 2005.
[12] Z. Wang and J. Crowcroft. Quality-of-service routing for supporting multimedia applications. *IEEE JSAC*, 14(7):1228–1234, Sept. 1996.

APPENDIX

We consider the problem of computing the trade-off curve of a bicriterion linear program

$$\begin{array}{ll} \text{minimize (w.r.t. } \mathbb{R}^2_+) & (c^T x, d^T x) \\ \text{subject to} & Ax = b, \quad x \succeq 0 \end{array}$$

where $A \in \mathbb{R}^{m\times n}$ with $\mathbf{rank}(A) = m$. We assume that all the extreme points of the feasible set are nondegenerate and we refer to the book [7] of Roos *et al.* for a careful treatment of parametric linear programming in the presence of degeneracy. A feasible x is an extreme point if the submatrix A_I of A formed by the columns indexed by $I = \{j \mid x_j > 0\}$ has full rank ($\mathbf{rank}(A_I) = \#I$). An extreme point is nondegenerate if it has exactly m positive components, *i.e.*, A_I is square and nonsingular. It follows that a nondegenerate extreme point x is uniquely determined by its *basis* I: we have

$$x_I = A_I^{-1} b, \qquad x_j = 0, \quad j \notin I.$$

We will also assume that the two objectives $c^T x$ and $d^T x$ are bounded below on the feasible set. This is true, for example, if c and d have nonnegative coefficients.

The trade-off curve is a convex piecewise-linear function obtained by plotting $d^T x(\lambda)$ versus $c^T x(\lambda)$ for $\lambda \geq 0$, where $x(\lambda)$ is an optimal solution of the linear program (LP)

$$\begin{array}{ll} \text{minimize} & c^T x + \lambda d^T x \\ \text{subject to} & Ax = b, \quad x \succeq 0. \end{array} \tag{8}$$

The dual of the LP (8) is

$$\begin{array}{ll} \text{maximize} & b^T y \\ \text{subject to} & A^T y \preceq c + \lambda d. \end{array} \tag{9}$$

A. The trade-off curve of the relaxed problem

The following parametric linear programming algorithm (PLPA) generates a sequence $\lambda^{(k)}$ with

$$0 = \lambda^{(0)} \leq \lambda^{(1)} \leq \cdots \leq \lambda^{(K)}$$

and a sequence of extreme points $x^{(k)}$ such that $x^{(k)}$ is optimal for (8) with $\lambda = \lambda^{(k)}$. The trade-off curve is then obtained by connecting $(c^T x^{(k)}, d^T x^{(k)})$ for $k = 0, \ldots, K$.

Compute $x^{(0)}$, an extreme optimal solution of LP (8) with $\lambda = 0$.

Repeat the following steps for $k = 0, 1, \ldots$.

1) *Compute* $\lambda^{(k+1)}$. Define

$$\lambda^+ = \sup\{\lambda \mid a_j^T A_I^{-T}(c_I + \lambda d_I) \leq c_j + \lambda d_j,\ j \notin I\} \tag{10}$$

where $I = \{i \mid x_i^{(k)} > 0\}$ and a_j is the jth column of A. If $\lambda^+ = \infty$ we terminate the algorithm with $K = k$. Otherwise, we take $\lambda^{(k+1)} = \lambda^+$.

2) *Compute* $x^{(k+1)}$. Choose an index $j \notin I$ for which $a_j^T A_I^{-T}(c_I + \lambda d_I) > c_j + \lambda d_j$ for $\lambda > \lambda^{(k+1)}$. Define $x^{(k+1)}$ as

$$\begin{aligned} x_I^{(k+1)} &= x_I^{(k)} - \hat{t} A_I^{-1} a_j, \\ x_j^{(k+1)} &= \hat{t}, \\ x_l^{(k+1)} &= 0, \quad l \notin I \cup \{j\}, \end{aligned}$$

where $\hat{t} = \sup\{t \mid x_I^{(k)} - t A_I^{-1} a_j \succeq 0\}$.

B. Discussion

We first note that $x^{(0)}$ exists since we assumed that the objective function $c^T x$ is bounded below over the feasible set.

Step 1 is based on the fact that $x^{(k)}$ is an optimal point of (8) for $\lambda^{(k)} \leq \lambda < \lambda^+$ (or $\lambda^{(k)} \leq \lambda \leq \lambda^+$ if λ^+ is finite). This can be seen as follows. The optimality conditions for the pair of LPs (8) and (9) are:

$$\begin{aligned} Ax = b, \quad x \succeq 0, \\ A^T y \preceq c + \lambda d, \\ x_i(c_i + \lambda d_i - a_i^T y) = 0, \quad i = 1, \ldots, n. \end{aligned}$$

Given that $x^{(k)}$ is optimal for $\lambda = \lambda^{(k)}$ and that $x_I^{(k)} \succ 0$, the optimality conditions at $\lambda = \lambda^{(k)}$ reduce to

$$A_I^T y = c_I + \lambda^{(k)} d_I, \qquad a_j^T y \leq c_j + \lambda^{(k)} d_j, \quad j \notin I.$$

A_I is square and nonsingular because $x^{(k)}$ is nondegenerate, so from the first condition we can uniquely determine the optimal y:

$$y = A_I^{-T}(c_I + \lambda^{(k)} d_I).$$

Since y must be dual feasible, we also have

$$a_j^T A_I^{-T}(c_I + \lambda^{(k)} d_I) \leq c_j + \lambda^{(k)} d_j, \quad j \notin I.$$

Next we note that for $\lambda \geq \lambda^{(k)}$,

$$y(\lambda) = A_I^{-T}(c_I + \lambda d_I)$$

satisfies the dual inequalities indexed by I with equality, so $y(\lambda)$ is feasible for all λ that satisfy

$$a_j^T y(\lambda) \leq c_j + \lambda d_j, \quad j \notin I.$$

This condition defines an interval for λ that includes $\lambda^{(k)}$. It holds for $\lambda \in [\lambda^{(k)}, \infty)$ if $\lambda^+ = \infty$ and for $\lambda \in [\lambda^{(k)}, \lambda^+]$ otherwise. The end point $\lambda^{(+)}$ is determined via a *minimum ratio test*:

$$\lambda^+ = \min_{j \in J} \frac{c_j - a_j^T A_I^{-T} c_I}{a_j^T A_I^{-T} d_I - d_j}, \qquad J = \{j \notin I \mid a_j^T A_I^{-T} d_I > d_j\}. \tag{11}$$

So we know that $y(\lambda)$ is dual feasible for all $\lambda \in [\lambda^{(k)}, \lambda^{(k+1)}]$. Since $x^{(k)}$ satisfies the complementary slackness conditions with $y(\lambda)$, we can conclude that $x^{(k)}$ and $y(\lambda)$ are primal and dual optimal for $\lambda^{(k)} \leq \lambda \leq \lambda^{(k+1)}$.

In step 2 we start by selecting an index $j \notin I$ such that $a_j^T y(\lambda) > c_j + \lambda d_j$ for $\lambda > \lambda^{(k+1)}$, *i.e.*, j is an index at which the minimum in (11) is achieved. We then determine

$$\hat{t} = \sup\{t \mid x_I^{(k)} - t A_I^{-1} a_j \succeq 0\}.$$

This value can be determined by another minimum ratio test. Define $v \in \mathbb{R}^n$ as

$$v_I = -A_I^{-T} a_j, \qquad v_j = 1, \qquad v_l = 0, \quad l \notin I \cup \{j\}.$$

Then

$$\hat{t} = \sup\{t \mid x^{(k)} + tv \succeq 0\} = \min_{i \in I, v_i < 0} \frac{-x_i^{(k)}}{v_i}. \tag{12}$$

We first show that $\hat{t}$ is finite. $\hat{t} = \infty$ means that $v = A_I^{-1} a_j \succeq 0$, and therefore that $x = x^{(k)} + tv$ is primal feasible for all $t \geq 0$. However

$$\begin{aligned} (c + \lambda d)^T (x^{(k)} + tv) &= (c + \lambda d)^T x^{(k)} + t(c_j + \lambda d_j - \\ &\qquad t(c_I + \lambda d_I)^T A_I^{-1} a_j) \\ &= (c + \lambda d)^T x^{(k)} + t(c_j + \lambda d_j - a_j^T y(\lambda)) \end{aligned} \tag{13}$$

and since the coefficient of t is negative for $\lambda > \lambda^{(k)}$, this means that for $\lambda > \lambda^{(k)}$, the cost function $(x + \lambda d)^T x$ is unbounded below along the ray $\{x^{(k)} + tv \mid t \geq 0\}$. This contradicts our assumption that the cost functions are bounded on the feasible set.

We also must have $\hat{t} > 0$: $\hat{t} = 0$ would mean that some of the coefficients of $x_I^{(k)}$ are zero, which contradicts the assumption that $x^{(k)}$ is a nondegenerate extreme point.

Next we show that $x^{(k+1)}$ is an extreme point. Select an index $i \in I$ for which $x_i^{(k+1)} = 0$, *i.e.*, i is one of the indices at which the minimum in (12) is attained. We have $v_i < 0$ and, since

$$A_I v_I = -a_j,$$

the columns of A indexed by $(I \cup \{j\}) \setminus \{i\}$ are linearly independent. Therefore $x^{(k+1)}$ is an extreme point. Since, by

assumption, all extreme points are nondegenerate, we know there is exactly one $i \in I$ for which $x_i^{(k+1)} = 0$.

Finally, we note from (13) that

$$(c + \lambda d)^T x^{(k+1)} < (c + \lambda d)^T x^{(k)} \tag{14}$$

for $\lambda > \hat{\lambda}^{(k)}$. This guarantees that the algorithm does not cycle, *i.e.*, the points $x^{(k)}$ are all distinct.