Some approaches to solving a multihour broadband network capacity design problem with single-path routing

D. Medhi [a,*] and D. Tipper [b]

[a] *Computer Science Telecommunications, University of Missouri-Kansas City, 5100 Rockhill Road, Kansas City, MO 64110, USA*

[b] *Department of Information Science and Telecommunications, University of Pittsburgh, Pittsburgh, PA 15260, USA*

Received July 1996; in final form August 1999

In this paper, we consider solution approaches to a multihour combined capacity design and routing problem which arises in the design of dynamically reconfigurable broadband communication networks that uses the virtual path concept. We present a comparative evaluation of four approaches, namely: a genetic algorithm, a Lagrangian relaxation based subgradient optimization method, a generalized proximal point algorithm with subgradient optimization, and, finally, a hybrid approach where the subgradient based method is combined with a genetic algorithm. Our computational experience on a set of test problems of varying network sizes services) shows that the hybrid approach often is the desirable choice in obtaining the minimum cost network while the genetic algorithm based approach has the most difficulty in solving large scale problems.

1. Introduction

Multihour network design for various communication networks, especially for circuit-switched networks, has been addressed by several researchers [3,12,14,16,25, 37,38,45,46,49] over the years. In multihour models, network traffic is considered for different hours during the day to reflect within the day load variation; this is done especially to take advantage of noncoincidence of busy hour (peak) loads between different node pairs in the network. Multihour network dimensioning (sizing) involves determination of network capacity at minimum cost that can meet the traffic demand at any time during the day by considering multihour traffic. Some of the work listed above considers network dimensioning under fixed routing while others have addressed the combined capacity and routing design problem. Several papers have addressed combined routing and capacity design problems for data networks considering only "single hour" load models, for example, see [21,22,24,43]; while a multitime period model has been addressed by [13]. The reader may note that

* Supported in part by the University of Missouri Research Board Grant K-3-40605 and by the NSF grant CDA-9422092.

multitime period problems are different from multihour problems; multitime period models typically consider capacity expansion problems over a period of time (years) while addressing issues such as discounting and capacity deferral (see, for example, [13,36,51,53,54]).

The specific problem we address here arises as a *part* of design of a dynamically reconfigurable ATM-based (asynchronous transfer mode) broadband networks presented in [38] that uses the concept of virtual path and virtual path switching [11,30,35]. A virtual path in ATM-based networks [52] can be defined to group a bundle of virtual circuits together (which may be statistically multiplexed) for ease of manageability and to reduce connection set up time for the virtual circuits; the reader is directed to [2,7,38,39,50] for further material on the flexibility of the virtual path concept for broadband network management. Thus, a virtual path is desirable between origin and destination nodes to bundle all connection requests for this node pair; on the other hand, separate virtual paths may be desirable for different service classes. Another network dynamic is that traffic demand changes with time of day; thus taking advantage of the concept of virtual path reconfigrability, it is desirable to capture the dynamic reconfiguration of virtual paths (and bandwidth allocated to them) that takes the traffic dynamics into consideration. We assume that the underlying physical topology (i.e., the nodes and the links) is given; the network design problem here is to determine modular capacity for each network link that minimizes the total network link cost so that the demand variation (given over multiple load hours) are met in the network by choice of different virtual path at different time of the day; see [38] for detailed discuss on the network design problem.

The purpose of this paper is to consider some solution approaches to this network design problem, and discuss benefits/advantages, if any, of one approach over another. Our first approach is based on a genetic algorithm [26] which has drawn attention in recent years for solving various combinatorial optimization problems. In this approach, we show how problem (P) can be replaced by an equivalent problem, and a proper coding rule can be used so that we have an environment suitable for using a genetic algorithm. This approach of problem equivalency and coding for communication network design problems using a genetic algorithm is particularly novel. It may be noted that genetic algorithms have been recently used for the topological design of local area networks [15]. The second approach is based on duality using subgradient optimization as presented in [38]; the duality based approach has been successfully used by several researchers for solving various data network design problems (see, for example, [12,13,21,22,34]). The third approach is a relatively new approach where we employ a generalized proximal point algorithm (GPPA)-based approach. The GPPA has been developed for solving generalized equations [27] and convex optimization problems [40]; in our case, to solve model (P), we combine the generalized proxima point concept with the use of duality and subgradient optimization. Finally, we consider a hybrid approach where both a genetic algorithm and duality based sub-

gradient optimization is combined. We then provide computational results for test networks that are extracted from real networks as well as from randomly generated networks and data to comparatively evaluate the performance of these approaches.

The rest of the paper is organized as follows. The optimization model for the network design problem is stated in section 2. In section 3, we present four algorithms. Computational results and merits/demerits of these algorithms are discussed in section 4. We briefly mention the following notation used in the rest of the paper: $\#(\cdot)$ denotes the cardinality of a set; $\|\cdot\|$ represents the Euclidean norm of a vector; $\lceil x \rceil$ represents the ceiling function (i.e., rounds x up to the nearest integer value).

2. Problem formulation

This model arises as a *part* of design of a dynamically reconfigurable ATM-based (asynchronous transfer mode) broadband networks presented in [38]. In this design problem, the path selection is for selection of a virtual path for a traffic pair between the origin and the destination node; it may be noted that different services classes may take different virtual paths for the same origin–destination pair in a certain load period while a single-virtual path may be defined for one service class to carry all the virtual circuits for that class for the same origin–destination pair. We will use the following notations:

$\mathcal{K}$ set of traffic demand node pairs in the network,

$\mathcal{S}$ set of traffic (service) types,

$\mathcal{L}$ set of possible links in the network,

$\mathcal{H}$ set of load hours (periods) during the day,

$\mathcal{J}_k^{sh}$ set of possible candidate paths for traffic type $s \in \mathcal{S}$, demand pair $k \in \mathcal{K}$ used for all $h \in \mathcal{H}$,

x_{kj}^{sh} path routing variables: 1 if traffic type $s \in \mathcal{S}_k$, $k \in \mathcal{K}$ uses path $j \in \mathcal{J}_k^{sh}$ in $h \in \mathcal{H}$; 0 otherwise,

a_k^{sh} traffic demand for traffic type $s \in \mathcal{S}$ for $k \in \mathcal{K}$ in $h \in \mathcal{H}$,

$\delta_{kj}^{s\ell h}$ link-path indicator: 1 if path $j \in \mathcal{J}_k^{sh}$ for traffic type $s \in \mathcal{S}$ and for pair $k \in \mathcal{K}$ in $h \in \mathcal{H}$ uses link $\ell \in \mathcal{L}$; 0 otherwise,

y_ℓ number of units of capacity needed on traffic link $\ell \in \mathcal{L}$ (capacity unit variable),

γ capacity of a high-speed network link unit,

c_ℓ cost of a high-speed network link unit on link $\ell \in \mathcal{L}$.

The multihour capacity design of broadband networks with single-path virtual path routing can be stated as follows [38].

Problem (P).

$$v_{\mathrm{P}} = \min_{\{x,y\}} \sum_{\ell \in \mathcal{L}} c_\ell y_\ell \tag{1}$$

subject to

$$\sum_{j \in \mathcal{J}_k^{sh}} x_{kj}^{sh} = 1, \qquad s \in \mathcal{S},\ k \in \mathcal{K},\ h \in \mathcal{H}, \tag{2}$$

$$\sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}} a_k^{sh} \sum_{j \in \mathcal{J}_k^{sh}} \delta_{kj}^{s\ell h} x_{kj}^{sh} \leqslant \gamma y_\ell, \qquad \ell \in \mathcal{L},\ h \in \mathcal{H}, \tag{3}$$

$$x_{kj}^{sh} = 0 \text{ or } 1, \qquad j \in \mathcal{J}_k^{sh},\ s \in \mathcal{S},\ k \in \mathcal{K},\ h \in \mathcal{H}, \tag{4}$$

$$y_\ell \geqslant 0 \text{ and integer}, \qquad \ell \in \mathcal{L}. \tag{5}$$

The problem defined by equations (1)–(5), will be referred to as problem (P). In problem (P), the objective function (1) represents the total network capacity cost over all links which is to be minimized. Constraints (2) and (4) are the decision of choosing a path (out of several possible paths) for a traffic type for a node pair at different times during the day to route all the associated demand a_k^{sh}, i.e., this is a "single route selection" or "nonbifurcated routing" case (the reader is referred to [38] for determination of traffic demand a_k^{sh} for a given offered load and quality-of-service parameter). The left-hand side of constraint (3) is the total link flow on link ℓ in hour h due to routing of demand a_k^{sh} by different traffic demand pairs and services in h. The right-hand side of (3) says that y_ℓ units of capacity is required to be provided on link ℓ so that the total link flow is satisfied for any of the load hours by the total bandwidth on the link γy_ℓ. Finally, (5) specifies that the capacity variables take only integral values.

Different sets of candidate virtual paths for each load period and traffic type for each node pair may be explicitly provided through the notation $\mathcal{J}_k^{sh}$. Thus, this model takes possible candidate paths as an input and uses the arc-path formulation [41]. Usually, a limit is put on the maximum number of links (for a virtual path) to reduce the number of intermediate virtual path switching nodes allowed for a virtual path set up. Hence, the arc-path formulation than a node-arc formulation is more appropriate in this scenario. path based broadband networks. Another example from communication networks where arc-path formulation is appealing, is in dynamic call routing teletraffic networks where at most two links are allowed to connect a call [3,25].

Note that this is a combined routing and capacity design model with discrete variables where we are to choose a path (different ones at different time during the day and for different traffic types) so that the total network capacity required is optimized. Further note that this model has only link cost, no explicit routing cost; this is often the case for planned demand servicing phase [3,4,25] of large telecommunications networks. Finally, note that this formulation consider nonsimultaneous as well as simultaneous multicommodity flow for the design problem by considering the load at different times during the day.

We have stated at the beginning that network design problems has been studied by various researchers; some of them do use arc-path formulations for multicommodity flow problems arising in such problems. Further, it may be noted that multicommodity flow "routing only" problem (either for discrete or for continuous variables) using arc-path approach in the context of communications networks as well as transportation networks has been addressed by several researchers over the years; see, for example, [5,9,10,17–20,23,33,34]. To our knowledge, the combinatorial optimization problem as presented in model (P), i.e., the multihour combined routing and capacity design problem (with single route selection case) in a multi-service environment has not received much attention in the literature.

3. Various approaches

In this work, we consider four approaches to solving design problem (P): the first one is based on mapping (P) to an equivalent problem to solve using the genetic algorithm-based approach (labeled GA); the second approach takes the Lagrangian relaxation-based dual subgradient approach where the constraint (3) is relaxed (labeled SGOPT); in the third approach, we consider an additional proximal point term only for capacity variables and then combine the generalized proximal point algorithm with subgradient optimization (labeled SG-GPP); finally, in the fourth approach, we take a hybrid approach where we start with SGOPT or SG-GPP to obtain the best solution possible which is then fed to the genetic algorithm approach as one of the chromosome in the initial population – this approach is labeled HYB. In the following, we discuss each of them in details.

3.1. Algorithm I: GA

In the first approach to solve problem (P), we consider a genetic algorithm-based [26] approach. Genetic algorithms have gained considerable attention in recent years for solving various combinatorial optimization problems (see, for example, [1,29,42,44, 48], and references therein). A genetic algorithm works on the concept of maximizing a fitness function by considering several generations of population which go through certain transformation from one generation to the next. Each population consists of a set of chromosome sequences. The fitness function is evaluated for each chromosome sequence (which takes only 0 or 1 values) to obtain the highest fitness value for that population. Typically, a genetic algorithm has the following steps [26]: generation of an initial population, selection of a new population from the given population by performing a recombination step to do cross-over and mutation, and the evaluation of the new population to observe improvement in the fitness function value. This process is continued over a finite number of generations of population and the chromosome sequence with the maximum fitness value over the entire generation is accepted as the solution.

To be able to develop a genetic algorithm for problem (P), we are first required to address two important issues:

(1) a fitness function that returns non-negative value that should be set in such a way that the fitness function is maximized,

(2) a coding scheme for a chromosome sequence.

For the first issue, we consider the following equivalent optimization problem (GA_P) of problem (P):

$$\max_{\{x\}}\left\{ f(x) = M - \sum_{\ell\in\mathcal{L}} c_\ell \left\lceil \max_{h\in\mathcal{H}}\left\{ \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k}\sum_{j\in\mathcal{J}_k^{sh}} \delta_{kj}^{s\ell h} a_k^{sh} x_{kj}^{sh} \right\} \Big/ \gamma \right\rceil \right\} \tag{6}$$

subject to

$$\sum_{j\in\mathcal{J}_k^{sh}} x_{kj}^{sh} = 1, \quad s\in\mathcal{S}_k,\ k\in\mathcal{K},\ h\in\mathcal{H}, \tag{7}$$

$$x_{kj}^{sh} = 0 \text{ or } 1, \quad j\in\mathcal{J}_k^{sh},\ s\in\mathcal{S}_k,\ k\in\mathcal{K},\ h\in\mathcal{H}, \tag{8}$$

where M is a large positive number such that the objective function $f(x)$ given in (6) is always non-negative (see appendix A for equivalency between (P) and (GA_P)). We use $f(x)$ as defined in (6) as the fitness function for the genetic algorithm.

For the second issue (coding), as a naive rule, a straight coding can be assigning a bit for every decision variable in a chromosome sequence (since they take the values 0 or 1). However, this will not only require a large bit sequence but may also *often* generate illegal sequences violating constraint (7). In our approach, we use a coding rule that exploits the problem structure given in the arc-path formulation. Notice that for each s, k, h we need to choose one route from a set of candidate paths. Thus, we do bit coding for the number of paths for each s, k, h , i.e., if $\mathcal{J}_k^{sh}$ is the set of paths, then the number of bits required is $\lceil \log_2 \#(\mathcal{J}_k^{sh})\rceil$ for coding all the paths for this s, k, h. Note that each bit sequence (within a specific s, k, h) can be mapped to a path number if $\#(\mathcal{J}_k^{sh})$ is exactly a power of two. However, the number of possible paths generated as an input to problem (P) may not always be a power of two; in such a situation it is possible to have a bit sequence that corresponds to a path number that is "illegal" in which case the fitness function, $f(x)$, is set to return a cost value of zero (analogous of infinite penalty in minimization). The bit sequence for each s, k, h is then concatenated to form a chromosome sequence. If we denote $S = \#(\mathcal{S})$, $H = \#(\mathcal{H})$, $K = \#(\mathcal{K})$, and assume each pair to have the same number of paths q, then the total bits required in a chromosome sequence is $SHK\lceil \log_2 q\rceil$. Overall, this scheme requires log-order less number of bits in the chromosome sequence than the "naive" rule (which requires $SHKq$ bits); also, the number of illegal sequences can be reduced or totally eliminated by properly choosing the number of candidate paths generated.

Having resolved the issues of the fitness function and the coding scheme, we now turn to the execution of the genetic algorithm for problem (GA_P). In our approach, the initial population is generated randomly (except for a sample, see section 3).

This, for our specific problem, means that each randomly generated sample in the initial population is obtained by randomly generating a path as an initial choice for each s, k, h to create an initial chromosome sequence. The selection phase involves choosing samples from the present population that will be the parents of the next generation. In the selection phase, typically a "weighted random roulette" strategy (rule-RR) [26,44,48] is used for selection of a pair of chromosomes (parents) for mating (i.e., chromosome n of the present population $\mathcal{M}$ is selected with a probability $f_n / \sum_{m \in \mathcal{M}} f_m$, where f_m is the fitness value of chromosome m). Besides using this rule, we have also considered the queen-bee strategy, due to [48]. In this strategy (Rule-QB), one parent for mating is always the chromosome with the best fitness function so far (i.e., the routing/capacity configuration that has the minimum cost so far) while the other one is obtained using the weighted random roulette rule.

A probability value p_{cross} is used in the determination of the cross-over point for the two parents selected. This, in our context, roughly corresponds to finding a cross-over point such that services and load hours for a certain number of traffic pairs from parents A and B (obtained from the selection process) are considered for creation of children A and B, respectively; for the rest of the traffic pairs (services, load hours) child A is obtained from parent B and child B from parent A, respectively. In obtaining the children, mutation is performed using a low mutation probability, p_{mute}, for each bit. In our case, for each s, k, h, this corresponds to changing a path from it present choice with a probability $1 - (1 - p_{\text{mute}})^{\lceil \log_2 q \rceil}$. This way a new generation is obtained for which the best fitness function is obtained. This process is continued for a certain number of generations and at the end the "layout" with the best fitness function which corresponds the lowest cost in (1) is obtained. A difficulty with using this genetic algorithm is that no information can be obtained about how far is the best solution obtained from the actual optimal solution. This can be addressed via duality discussed in the next section.

3.2. Algorithm II: SGOPT

Lagrangian relaxation with subgradient optimization has been used successfully by several researchers for solving various communication network routing, or combined routing and capacity design problems (see, for example, [12,13,20–23,34]). This approach has been described for problem (P) in [38]. We restate here for the sake of completeness and also since some of the same notations will be used in presenting SG-GPP algorithm in section 3.3. Convergence results for subgradient optimization can be found, for example, in [41]. The essence of this decomposition algorithm is to consider the dual problem of (P) by relaxing constraints (in this case (3)) to arrive at simpler subproblems which can be solved easily, and then use updating of dual variables to solve simpler subproblems iteratively to drive towards the optimal solution of the original problem (P); these subproblems correspond to solving decoupled routing and link capacity problems. A difficulty with the subproblems corresponding to capacity variables is that it requires an upper bound on the values of the capacity

variables; its use will be seen specifically in (17). Any sufficiently large number that keeps the problem feasible can be used for the upper bound. Thus, we start with adding an artificial upper bound on the variables y. Specifically, let the bound be $\hat{y}_\ell$, $\ell \in \mathcal{L}$, i.e.,

$$y_\ell \leqslant \hat{y}_\ell, \quad \ell \in \mathcal{L}, \tag{9}$$

and in compact vector form, $y \leqslant \hat{y}$. Let $u = (u_\ell^h)$ be the dual multiplier associated with the constraint (3). The associated Lagrangian is

$$L(x,y,u) = \sum_{\ell\in\mathcal{L}} c_\ell y_\ell + \sum_{\ell\in\mathcal{L}} \sum_{h\in\mathcal{H}} u_\ell^h \left(\sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}} a_k^{sh} \sum_{j\in\mathcal{J}_k^{sh}} \delta_{kj}^{s\ell h} x_{kj}^{sh} - \gamma y_\ell \right). \tag{10}$$

Rearranging, $L(x,y,u)$ yields

$$L(x,y,u) = \sum_{\ell\in\mathcal{L}} \left(c_\ell - \gamma \sum_{h\in\mathcal{H}} u_\ell^h \right) y_\ell + \sum_{h\in\mathcal{H}} \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}} a_k^{sh} \sum_{j\in\mathcal{J}_k^{sh}} \sum_{\ell\in\mathcal{L}} u_\ell^h \delta_{kj}^{s\ell h} x_{kj}^{sh}. \tag{11}$$

The dual problem (D) is then defined as

$$v_{\mathrm{D}} = \max_{u\geqslant 0} g(u), \tag{12}$$

where

$$g(u) = \min_{x,y} L(x,y,u). \tag{13}$$

Note that for given u, the Lagrangian is separable in x and y and reduces (11) to solving two independent subproblems, i.e.,

$$\min_{x,y} L(x,y,u) = \min_y L_1(y,u) + \min_x L_2(x,u) = g_1(u) + g_2(u), \tag{14}$$

where

$$g_1(u) = \min_y L_1(y,u) = \min_y \left\{ \sum_{\ell\in\mathcal{L}} \left(c_\ell - \gamma \sum_{h\in\mathcal{H}} u_\ell^h \right) y_\ell \mid 0 \leqslant y \leqslant \hat{y} \right\}, \tag{15}$$

and

$$\begin{aligned} g_2(u) &= \min_x L_2(x,u) \\ &= \min_x \left\{ \sum_{h\in\mathcal{H}} \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}} a_k^{sh} \sum_{j\in\mathcal{J}_k^{sh}} \left(\sum_{\ell\in\mathcal{L}} u_\ell^h \delta_{kj}^{s\ell h} \right) x_{kj}^{sh} \,\middle|\, \sum_{j\in\mathcal{J}_k^{sh}} x_{kj}^{sh} = 1,\ s\in\mathcal{S},\ h\in\mathcal{H}, \right. \\ &\qquad \left. k\in\mathcal{K};\ x_{kj}^{sh} = 0 \text{ or } 1,\ j\in\mathcal{J}_k^{sh},\ s\in\mathcal{S},\ k\in\mathcal{K},\ h\in\mathcal{H} \right\}. \end{aligned} \tag{16}$$

For subproblem (15), observe that $L_1(y,u)$ is further separable to each link variable since it has only bounding constraints on the link variables, and thus the solution to (15) for each ℓ can be easily obtained by setting

$$y_\ell^*(u) = \begin{cases} 0, & \text{if } c_\ell \geqslant \gamma \sum_{h\in\mathcal{H}} u_\ell^h, \\ \hat{y}_\ell, & \text{if } c_\ell < \gamma \sum_{h\in\mathcal{H}} u_\ell^h. \end{cases} \tag{17}$$

The other subproblem (16) is also separable for each h, k and s since there is no dependency constraint among h, k and s. The solution is easily obtainable by setting the variable for the appropriate path j to 1 for which the "path cost" $\sum_{\ell\in\mathcal{L}} u_\ell^h \delta_{kj}^{s\ell h}$ is the least among all the paths for that specific h, k, s. We denote this path index by j^* so that $x_{kj^*}^{sh}(u) = 1$.

Since the objective function in dual problem (D) is nondifferentiable, a subgradient approach is used to solve the dual problem [28]. This method iterates on the dual variable u. Thus, given u, once the solutions to the subproblems (15) and (16) are obtained, a dual subgradient, $\pi = (\pi_\ell^h)$, for $g(\cdot)$ is computed using

$$\pi_\ell^h = \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}} a_k^{sh} \delta_{kj^*}^{s\ell h} x_{kj^*}^{sh}(u) - \gamma y_\ell^*(u), \quad \ell \in \mathcal{L},\ h \in \mathcal{H}. \tag{18}$$

Then the dual multiplier u is updated using

$$u_\ell^h \leftarrow \max\{0, u_\ell^h + \lambda \pi_\ell^h\}, \quad \ell \in \mathcal{L},\ h \in \mathcal{H}, \tag{19}$$

where the step size λ is given by (see [28])

$$\lambda = \rho \frac{\tilde{g} - g(u)}{\|\pi\|^2}, \tag{20}$$

where $\tilde{g}$ is an upper bound on the dual objective and the relaxation parameter ρ is chosen such that $0 < \rho \leqslant 2$.

A primal feasible solution for problem (P) can be easily obtained at each iteration by considering the solution $x_{kj^*}^{sh}(u) = 1$ to compute the left-hand side of (3) so as to generate a feasible y, and in turn, we also can compute a primal objective function value. As the dual iteration progresses, we check for decreases in the primal objective value and store the best primal solution, and the best primal cost so far.

Note that if the integrality of link variables y is relaxed in the original problem (P), then we have a mixed integer programming (MIP) problem and it is clear that the optimal objective function value for this mixed integer problem, $v_{\text{MIP}} \leqslant v_{\text{P}}$. Coupling this fact with the result of the weak duality theorem [41, p. 203], we have the relation

$$v_{\text{P}} \geqslant v_{\text{MIP}} \geqslant v_{\text{D}}.$$

Given this observation, the upper bound on the dual objective, $\tilde{g}$, is set to be the lowest value of the objective function of the MIP problem as the dual iteration progresses. Note that using SGOPT, we cannot only get a primal solution but also can assess its quality by computing the duality gap.

3.3. Algorithm III: SG-GPP

In this approach, we combine Lagrangian relaxation based subgradient optimization with a generalized proximal point algorithm [27,40]. Note that in algorithm SGOPT, we have added an artificial bound (9) which is used in solving $g_1(u)$ (cf. (17)). Depending on the weight from summing up dual multipliers over h, the solution switches between 0 and $\hat{y}_\ell$, and does not provide any value between these two extremes. Since this solution is used in the computation of the subgradient which is, in turn, used as the direction for updating the dual iterate (see (19)), a "better" solution of $g_1(u)$ for each link ℓ is desirable in the hope of obtaining a better direction. Recently, Ha [27] has presented a generalized proximal point algorithm for solving generalized equations; this has been extended to convex optimization problems by Medhi and Ha [40]. In a regular proximal point algorithm (see, for example, [6,47]), a quadratic ("proximal point") term is added to the objective function for all variables while in the generalized proximal point (GPP) method, a quadratic term ("proximal point" term) is added only to a subset of variables. In the present problem, we employ the GPP idea where this quadratic term is added only to the capacity unit variables y_ℓ (*not* to the routing variables). It may be noted that in this approach we do not need to consider the upper bounding on y, (9), used in SGOPT; however, we need to introduce an outer iteration loop for GPP. We describe our approach below.

Consider a positive bounded sequence $\{\alpha(t)\}$ (referred as the proximal point parameter) which is nonincreasing as $t \to \infty$, where t is the counter for the outer (GPP) iteration. Suppose we are at the point $(x(t), y(t))$ at outer iteration t. We are interested in solving the following modified problem (P_M), parameterized in the outer GPP iteration t:

$$v_{\mathrm{P}}(t) = \min_{\{x,y\}} \sum_{\ell \in \mathcal{L}} c_\ell y_\ell + \frac{1}{2\alpha(t)} \big\| y - y(t) \big\|^2 \tag{21}$$

subject to (2)–(5).

The difference between (P) and (P_M) is only in the objective function where a quadratic term is added only for the link unit variables. The modified dual problem to (P_M) is

$$\max_{u \geqslant 0} \; g(t; u). \tag{22}$$

The corresponding $g_2(t; u)$ remains the same as in (16) since there is no explicit dependency on $x(t)$. However, we have the following $g_1(t; u)$ which is different from $g_1(u)$ as given in (15):

$$g_1(t; u) = \sum_{\ell \in \mathcal{L}} \min_{y_\ell} \left\{ \left(c_\ell - \gamma \sum_{h \in \mathcal{H}} u_\ell^h \right) y_\ell + \frac{1}{2\alpha(t)} \big(y_\ell - y_\ell(t) \big)^2 \;\Big|\; y_\ell \geqslant 0 \right\}. \tag{23}$$

Due to separability (as before), the minimization is required to be done only over each single, link variable independently. The solution for each ℓ is given by

$$y_\ell^*(t;u) = \max\left\{0,\ y_\ell(t) - \alpha(t)\left(c_\ell - \gamma \sum_{h\in\mathcal{H}} u_\ell^h\right)\right\}. \tag{24}$$

Now this solution for y along with the solution obtained from (16) for x can be used for computing the subgradient (18) and as such updating can be performed as in SGOPT for (inner) dual iteration. Thus, the inner iteration involves solving modified problem (P_M) using the dual problem via subgradient optimization.

Once the solution $(x(t+1), y(t+1))$ is obtained for problem (P_M), then the outer iteration t is updated and problem (P_M) is solved for new $(x(t), y(t))$ and $\alpha(t)$; usually, this process is continued under certain termination criteria and convergence results which is discussed in [40] for convex optimization problems. Since we are applying the GPP idea to a combinatorial optimization problem and not to a convex problem, we continue the outer iteration for a finite number of steps before we stop computation. Throughout the entire process, we continually use the solution from (16) to compute a feasible y_ℓ and in turn compute a primal cost for the original problem and store any solution if the cost has been lowered.

3.4. Algorithm IV: HYB

An important item to note from either algorithm SGOPT or algorithm SG-GPP is that the lowest primal solution obtained is dictated by the solution obtained from (16). On closer inspection, we observe that there is a limitation imposed by the "path cost" $\sum_{\ell\in\mathcal{L}} u_\ell^h \delta_{kj}^{s\ell h}$ if the same set of candidate paths were used by the different services for a specific demand pair in a specific load hour. That is, the cost coefficient corresponding to path index j is the same for all services for a specific k and h. Thus, the j^* obtained, i.e., the route selected by (16) for all services for a specific k and h is the same. Although, this is an acceptable solution that we have used in computing the lowest primal cost for a feasible primal solution, it can be better for different services to take different paths that may possibly further lower the primal objective cost (1) – this is already modeled in problem (P) which is not captured by the duality based approaches. This has motivated us to consider a hybrid approach where we first run SGOPT or SG-GPP followed by a run of the genetic algorithm since GA has no imposition on path choice among different services. In this hybrid approach, we consider the best solution obtained from either SGOPT or SG-GPP as one of the samples in the initial population (the rest of the population is generated as discussed in section 4) and then the genetic algorithm is executed. We refer to this approach as HYB.

4. Computational results

To make a comparative evaluation of the above algorithms, we have studied seven test problems of varying network sizes. For all test problems, we consider three load hours during the day, i.e., $H = 3$, and two traffic types, i.e., $S = 2$. The first three test problems R7, R10 and R15 (see figures 1–3) are extracted from real networks, including the traffic pattern at different times during the day for one traffic type (voice service) [37]. Here the number after R represents the number of traffic nodes in the

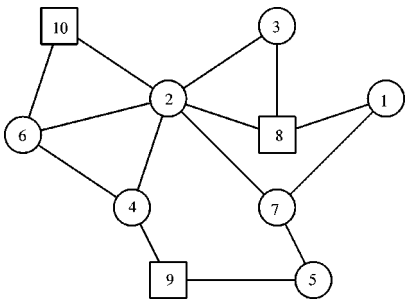

Figure 1. R7.

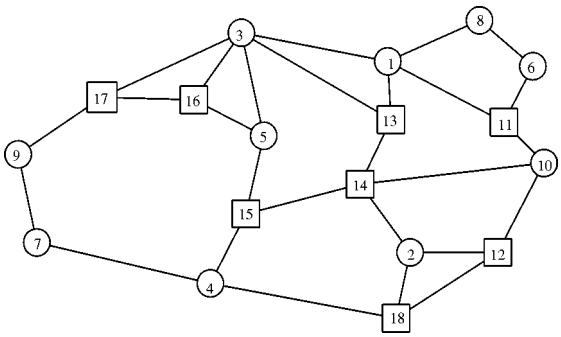

Figure 2. R10.

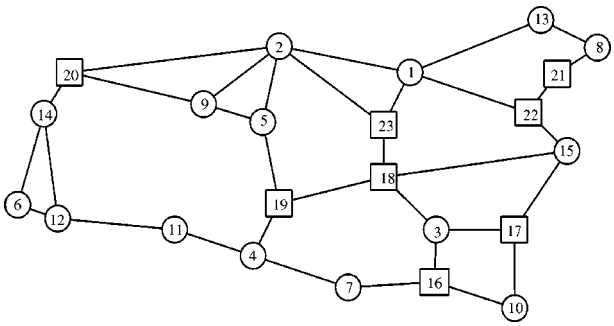

Figure 3. R15.

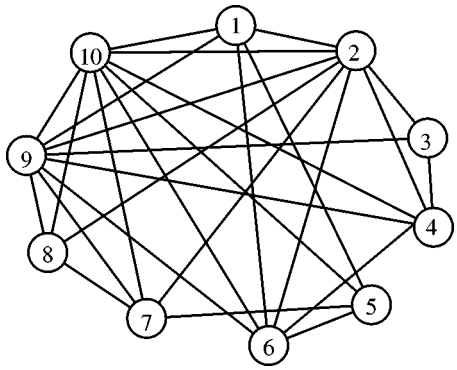

Figure 4. A10.

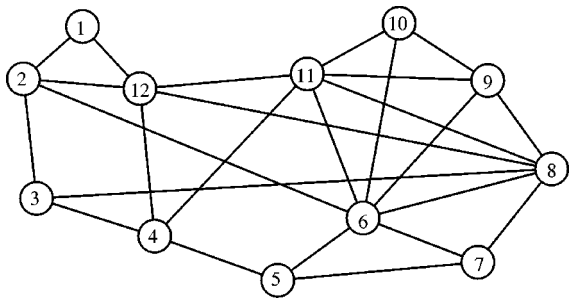

Figure 5. A12.

topology. Due to the nonavailability of measurement based traffic patterns for a second traffic type, the traffic load was uniformly generated for the second traffic class from the load for the first traffic class for these three problems. Originally the load was given in number of connections to be admitted which is mapped to a bandwidth requirement (see [38] for details on this mapping). Preserving the traffic pattern for different hours, the bandwidth requirement is scaled by the same factor so that the traffic demand (a_k^{sh}) as considered for problem (P) is given in Mega-bits per second (Mbps). (Note that each of these three test networks has some additional nodes known as ATM cross-connect nodes (marked with a square) than the traffic nodes which are ATM switching nodes (marked with a circle) – see figures 1–3; these cross-connect nodes should be understood as "routing only" nodes for ATM-virtual path set up through the cross-connect nodes for traffic for ATM switching node (demand) pairs and do not have any of their own exogenous traffic; see [38] for an explanation.) The topologies for problem A10, A12, A26 shown in figures 4–6, respectively, are drawn from previous related literature (see, for example, [34]); for these three test problems, the traffic data for both services types for three load hours and all traffic pairs in number of connections to be admitted is originally generated randomly from a uniform distribution which is then mapped to bandwidth requirements in Mbps. The topology for the final test problem A50 shown in figure 7 is a mixture of random generation and manual intervention to

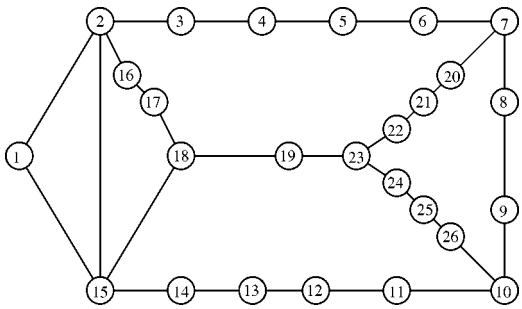

Figure 6. A26.

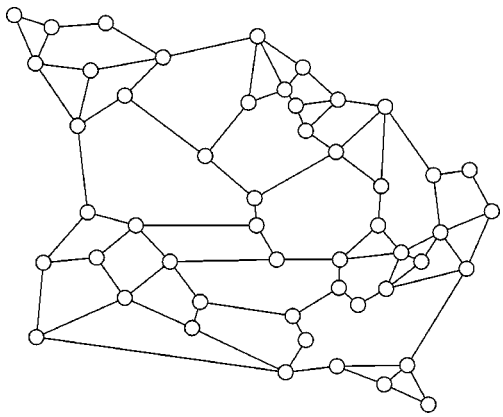

Figure 7. A50.

Table 1
Network topology information for example networks.

Network	No. of nodes	No. of traffic pairs $\#(\mathcal{K})$	No. of links $\#(\mathcal{L})$	No. of paths $\#(\mathcal{J}_k^{sh})$
R7	7	21	14	6
R10	10	45	27	10
R15	15	105	33	15
A10	10	45	28	7
A12	12	66	25	7
A26	26	325	30	8
A50	50	1225	82	8

provide connectivity. The traffic data for A50 was generated randomly, similar to A10, A12 and A26. A summary on topology information for the test problems is provided in table 1. The link unit γ for the link unit for all the test problems was assumed to be 155 Mbps.

Table 2
Parameter values used for genetic algorithm (total of sixteen settings).

Parameter names	Settings
Selection	Rule-RR, rule-QB
p_{cross}	0.60, 0.65
p_{mute}	0.01, 0.02
Random path selection	Bit-based, path-based

All the algorithms were implemented in C and the computational work was performed on a DEC Alpha AXP running OSF/1 operating system.[1] The compilation is performed using its native cc compiler with optimization option (O). The candidate paths for all the problems were generated using a k-shorted path algorithm based on distance [32]. The candidate path set generated for a traffic pair is used as the same for all load hours and services for this pair. The candidate path set is then provided as an input to the algorithms. Before we discuss the results, we first discuss implementation parameters used in the algorithms.

For the genetic algorithm, we have used a population size of 50 and the number of generations to be 100. We have run various parameter values as listed in table 2; thus, for each test problem we run a total of sixteen runs for various parameter setting values. The last option in table 2 indicates the way a chromosome sequence is generated in the initial population. In all runs, a sample in the initial population contains the bit sequence corresponding to the shortest-distance path obtained from the k-shortest path generation.

In SGOPT, we have set the maximum iterations to 800. The relaxation parameter ρ is started with 2 and is halved if the dual objective function value does not improve in forty iterations while restricting ρ to take a value not less than 0.0005. value is obtained by iteration 450, and thus, 800 is more than adequate. In SG-GPP, we have set the maximum for outer GPP iterations to five; the inner iteration maximum for subgradient optimization is set at 800 as in SGOPT; i.e., SGOPT phase is repeated five times with new $(x(t), y(t))$ and $\alpha(t)$.

In our work, we used two different values of c_ℓ. The unit cost of links of 155 Mbps consists of two components: a termination cost and a distance based cost. For our first set of run, we use 100 as the cost of each termination port (on each end) and 0.1 as the distance cost per mile. Thus, in this case, $c_\ell = 2 \times 100 + 0.1 \times D_\ell$, where D_ℓ is the distance in miles for 155 Mbps pipe for link ℓ. This indicates the growing trend in telecommunications networks where distance based cost can be small due to the deployment of fiber optic cables, while the termination cost is relatively more expensive. In this case, the termination cost is the dominant cost. The results obtained are listed in table 3. We have also run the same set of test problems with a different cost value where the termination cost is not as significant as the distance cost

[1] DEC Model 3000/300, 128 MB main memory, 256 KB cache, 150 MHz, SPECfp92 = 89, SPECint92 = 64.

Table 3
Cost of network by various algorithms and duality gap (for termination cost = 100.0).

	GA			SGOPT			SG-GPP			HYB		LPgap[3]
	Cost	%[1]	%[2]	Cost	%[1]	%[2]	Cost	%[1]	%[2]	Cost	%[2]	(%)
R7	14491.70	8.13	10.03	13402.00	0.00	1.75	13402.00	0.00	1.75	13171.10	0.00	9.58
R10	51434.50	3.99	4.80	49459.10	0.00	0.77	49459.10	0.00	0.77	49080.1	0.00	7.72
R15	84938.80	8.08	8.08	79772.00	1.50	1.50	78590.30	0.00	0.00	78590.30	0.00	7.47
A10	33954.00	0.00	0.00	36487.90	7.46	7.46	35592.60	4.83	4.83	34155.20	0.59	16.94
A12	57120.20	7.84	8.27	52968.00	0.00	0.40	53172.90	0.39	0.79	52756.70	0.00	11.92
A26	607622.60	9.83	9.83	553256.30	0.00	0.00	553425.70	0.03	0.03	553256.30	0.00	1.32
A50	2548611.70	5.48	5.48	2416316.50	0.00	0.00	2416316.50	0.00	0.00	2416316.50	0.00	3.87

[1]Percentage difference in cost with the lowest cost for the first three algorithms.
[2]Percentage difference in cost with the lowest cost for the four algorithms.
[3]Computed for the lowest cost (for all four algorithms) compared to the relaxed LP optimal cost obtained using CPLEX.

Table 4
Cost of network by various algorithms and duality gap (for termination cost = 10.0).

	GA			SGOPT			SG-GPP			HYB		LPgap[3]
	Cost	%[1]	%[2]	Cost	%[1]	%[2]	Cost	%[1]	%[2]	Cost	%[2]	(%)
R7	3205.00	2.01	3.68	3142.00	0.00	1.64	3142.00	0.00	1.64	3091.1	0.00	9.41
R10	14644.10	0.00	0.37	14816.20	1.18	1.55	14780.10	0.93	1.30	14590.50	0.00	6.14
R15	21850.30	1.80	1.80	21526.80	0.29	0.29	21463.90	0.00	0.00	21463.90	0.00	7.75
A10	3894.00	0.00	0.65	4077.70	4.72	5.39	3965.70	1.84	2.50	3868.8	0.00	16.51
A12	6607.60	8.16	9.16	6108.90	0.00	0.93	6168.50	0.98	1.91	6052.90	0.00	10.91
A26	65462.60	8.61	8.61	60378.40	0.17	0.17	60274.50	0.00	0.00	60274.50	0.00	1.22
A50	268551.70	5.78	5.78	253887.20	0.00	0.00	255596.50	0.67	0.67	253887.20	0.00	3.14

[1]Percentage difference in cost with the lowest cost for the first three algorithms.
[2]Percentage difference in cost with the lowest cost for the four algorithms.
[3]Computed for the lowest cost (for all four algorithms) compared to the relaxed LP optimal cost obtained using CPLEX.

to see if there is any impact on the results; in this case, we set the termination cost to 10.0 and the distance per mile cost to 0.1; i.e., $c_\ell = 2 \times 10 + 0.1 \times D_\ell$. The results obtained are presented in table 4.

We start with results presented in table 3 where we report the lowest network cost obtained by each of the algorithms. For SG-GPP in this case, we have started with α to be 10 which is reduced by half at each outer GPP iteration. The cost for GA reported here is for the lowest cost obtained from sixteen different runs. We also report relative network cost of each algorithm compared to the minimum of GA, SGOPT and SG-GPP; this is marked with note 1. Further, the relative network cost is reported compared to the minimum for all the algorithms in a separate column (marked note 2). Thus, the entry with 0.00% refers to the algorithm with the lowest cost.

First, we discuss results without considering HYB (for table 3). We observe that the cost obtained using GA is between 4% and 10% higher than the lowest cost except for problem A10. This suggests that, in general, "pure" GA had the most difficulty in lowering design cost; in fact, for the largest problems A26 and A50, GA could not generate a sequence with cost lower than the cost for the shortest-distance based rule provided as a sample in the initial population. We observe that SGOPT is nearly comparable to SG-GPP while SG-GPP providing somewhat better result (while in two problems SGOPT has higher cost by 1.5% and 2.5% than SG-GPP, in two other problems, SGOPT produced lower cost than SG-GPP by 0.4% and 0.03% only; for the rest of the problems, they produce the same lowest cost).

To run HYB, we have used the best solution from SGOPT or SG-GPP as a sample in the initial population; a second sample in this initial population is based on the shortest-distance path, and the rest are generated randomly based on a random path choice. Another set of run is done with this same initial population except for one sample where the random choice of the first two shortest paths is also included in the initial population. From our run of "pure" GA for sixteen settings, we have observed that usually rule-QB with p_{mute} value 0.01 provided the best result for p_{cross} value of either 0.65 or 0.60. Hence, we used rule-QB with p_{mute} value of 0.01 for both values of p_{cross} to run HYP for the two sets of the initial population just described. In tables, we show the lowest of the cost of these runs under the HYB column. It may be noted that only for problem R10 in table 3 (R7 and A10 in table 4), the second choice of the initial population produced lower cost.

Considering all four algorithms, it is interesting to observe that HYB was able to reduce the cost from the best of SGOPT and SG-GPP in four of seven problems. Even for problem A10, it brought down the cost to within 0.59% of the lowest cost obtained with "pure" GA (the only problem where GA did better). This shows that it is better to be able to take different routes by two services for the same traffic pair in a particular load period; this can indeed reduce network cost as opposed to all services for a node-pair taking the same path. We, however, remark that GA had the hardest time with the two largest problems A26 and A50 where GA could not improve at all on the initial lowest cost solution from the initial population (or from the initial population fed in the case of HYB). The situation did not improve even when we

Table 5
Computing time in seconds.

Network	GA[1]	SGOPT	SG-GPP	CPLEX[2]
R7	10.09	2.70	13.88	0.15
R10	35.07	19.02	95.83	0.88
R15	95.62	92.55	516.77	4.47
A10	23.05	6.93	34.92	0.47
A12	35.57	11.56	59.10	1.28
A26	394.75	218.30	1173.77	27.18
A50	2128.03	1219.68	6071.88	907.27

[1]Average per setting over sixteen settings.
[2]Relaxed LP solution time.

increased the population size from 50 to 80 and the number of of generations from 100 to 300; this suggests that GA-based approach may not be viable for large-scale problems.

In table 4, we report network cost when the dominant cost in c_ℓ is the distance cost (not the termination cost). We observe quite similar behavior as in table 3. In this case, GA had comparatively less difficulty with R7, R10 and R17. In four out of the seven test problems, SG-GPP did better than SGOPT while providing the same result in one of them. HYB was able to obtain the lowest cost among the four algorithms for the first five problems. (We note that for SG-GPP here, we use the starting α to be 100 for R7, R10 and R15 and 1000 for A10, A12, A26 and A50. This is to reflect the desire for a nominal initial cost contribution due to the proximal point term at the start of the computation.)

In table 5, we report computing time corresponding to table 3; timing is essentially the same for table 4 runs since the only difference is in the link cost c_ℓ. Observe that in all cases SGOPT took less computing time than a single run of GA while producing lower network cost (except for A10). As expected SG-GPP took about five time more computing time than SGOPT; this is not surprising since the maximum outer iteration was set to five. We observe that in most cases, SG-GPP was able to obtain the lowest cost in the first outer iteration (sometimes even providing lower cost than with SGOPT); only in one case it went over three. Thus, it seems a good rule of thumb may be to run two outer iterations of SG- GPP followed by GA for two parameter setting values (as done in HYB). If this directive is followed, we can estimate that HYB for the smallest problem R7 would take about 25 s while the largest problem A50 would take less than two hours.

Finally, we report our attempt to solve the test problems using the CPLEX solver [8]. The linear programming (LP) relaxed problem of problem (P) is the problem where all routing and capacity variables are relaxed to take real values – the computing times are shown in table 5 for the relaxed problems; for the smaller problems, the solution is obtained quickly while for the largest problem the relaxed LP solution is only 25% less than the SGOPT solution time. On the other hand, the mixed integer

programming model where routing variables were kept real while capacity variables were kept discrete was difficult for CPLEX MIP solver to solve – it could solve only problem R7 and could not solve the other problems even after running for more than an hour of cpu time for each of the problem. When all variables were set to be discrete, CPLEX MIP solver could *not* solve the smallest problem problem R7 even after running it for more than an hour of CPU time. Another heuristic to obtain an integral solution would be to use the optimal solution from the relaxed LP and then rounding for the path choices to obtain a heuristic solution; this heuristic when tried on problem R7 results in cost 14565.2 which is higher than all the methods reported in table 3 indicating the limitation of this heuristic. In tables 3 and 4, we show LPgap which is the gap between the lowest solution obtained among all the methods compared to the LP relaxed solution obtained used CPLEX. It may be noted that while the gap is not much in most cases, it is more than 15% for test problem A10. We have run this test problem with the number of paths increased to fifteen (from seven) to see if we missed any possible good paths in the input; however, the lowest cost obtained did not change at all. While the 15% gap may give the appearance that the solution quality is not good, it is however possible for the gap to be very high depending on problem parameters. In appendix B, we present a three-node example where the optimal integer solution leads to a duality gap of 566.67%!

5. Discussion

In this paper, we have addressed the multihour routing and capacity design problem (P) that arises in the design of dynamically reconfigurable broadband networks. For this combinatorial optimization problem, we have considered four approaches and have presented a comparative study. While Lagrangian relaxation based subgradient optimization has been used as the approach for data network design problem by several researchers, neither a genetic algorithm based approach nor a generalized proximal point algorithm coupled with subgradient optimization, nor a hybrid approach received much attention in current literature. Our computational results indicate that “pure” GA is the least effective method in almost all cases both in terms of computing network cost as well the computational time; further, it was totally ineffective for the two largest test problems, A26 and A50. SGOPT remains a viable approach while SG-GPP and, especially, HYB are new approaches that may be given serious consideration as alternative acceptable approaches. The relative success of HYB also suggests that using GA following SGOPT or SG-GPP is a good approach. In any case, we have made the observation that it is better to be able to take different routes by two services for the same traffic pair in a particular load period; this can indeed reduce network cost as opposed to all services for a node-pair taking the same path – this observation, although inherently present in the problem formulation (P), can be allusive to a casual observer and, thus, is brought to the attention of the reader.

Acknowledgements

The traffic data used in R7, R10 and R15 is based on data provided for another work [37] by Sprint Corporation and is greatly appreciated. The comments by the anonymous reviewers were extremely helpful in improving the content and the presentation of this paper.

Appendix A. Equivalency of problem (P) and problem (GA_P)

First, we note that (2) is the same as (7), and that (4) is the same as (8). Let

$$\left\lceil \max_{h\in\mathcal{H}}\left\{\sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k}\sum_{j\in\mathcal{J}_k^{sh}}\delta_{kj}^{s\ell h}a_k^{sh}x_{kj}^{sh}\right\}\Big/\gamma\right\rceil = z_\ell, \quad \ell\in\mathcal{L}. \tag{A.1}$$

Obviously, $z_\ell \geqslant 0$ and takes integral values, thus, is the same as (5). Now, the maximization of the objective function in (6) is transformed to

$$\max\left\{M - \sum_{\ell\in\mathcal{L}} c_\ell z_\ell\right\}.$$

This is equal to

$$M - \min\sum_{\ell\in\mathcal{L}} c_\ell z_\ell$$

which is equivalent to (1). Further, by definition, (A.1) satisfies the following:

$$\sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}_k}\sum_{j\in\mathcal{J}_k^{sh}}\delta_{kj}^{s\ell h}a_k^{sh}x_{kj}^{sh} \leqslant \gamma z_\ell, \quad h\in\mathcal{H},\ \ell\in\mathcal{L},$$

which is the same as the constraints (3).

Appendix B. A three-node example illustration

Consider a three-node network with a single service class and a single load hour (thus, we will ignore the subscripts h and s), and $\gamma = 1$. Assume that the traffic demand to be uniform for each node-pair and is given to be $a_k = a$ ($k = 1, 2, 3$). We also set $c_\ell = 1$. We are only interested in the value of a in the range $0 < a \leqslant 2/3$. For the relaxed LP problem (i.e., allowing capacity variables to take real values), the minimum cost is $3a$ (i.e., capacity a is assigned to each link). Now, assume that the capacity variables are integral. For the same demand set, we now need two of the three links to have one unit of capacity each, and the third link to be of zero capacity (since the demand for this end point can be routed over the other two links due to ample capacity availability); thus, the minimum optimal cost is 2 for this integer program. Note that this integer optimal cost is $100(2-3a)/(3a)\%$ more than the relaxed LP

optimal solution for a in the range $0 < a \leqslant 2/3$; this also then gives the best duality gap possible for the integer program. If we use a to be 0.6, then the duality gap is alteady 11.11%, on the other hand, if a is 0.1, the duality gap jumps to 566.67%! Thus, the duality gap is not always a good indicator of the quality of solution, especially when the traffic demand is low compared to the modular capacity of a link.

References

[1] E.J. Anderson and M.C. Ferris, Genetic algorithms for combinatorial optimization: The assembly line balancing problem, ORSA Journal on Computing 6 (1994) 161–173.

[2] A. Arvidsson, Management of reconfigurable virtual path networks, in: *Proc. of the 14th Internat. Teletraffic Congress*, Antibes, France (June 1994) pp. 931–940.

[3] G.R. Ash, R.H. Cardwell and R.P. Murray, Design and optimization of networks with dynamic routing, Bell Systems Technical Journal 60(8) (1981) 1787–1820.

[4] G.R. Ash, A.H. Kafker and K.R. Krishnan, Servicing and real-time control of networks with dynamic routing, Bell Systems Technical Journal 60 (1981) 1821–1845.

[5] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. (Prentice-Hall, Englewood Cliffs, NJ, 1992).

[6] D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods* (Prentice-Hall, Englewood Cliffs, NJ, 1989).

[7] J. Burgin, Broadband ISDN resource management, Computer Networks & ISDN Systems 20 (1990) 323–331.

[8] CPLEX linear optimizer 4.0.3 with mixed integer solver; available at: `www.cplex.com`.

[9] S.C. Dafermos, An extended traffic assignment model with application to two-way traffic, Transportation Science 5 (1971) 366–389.

[10] R. Dembo and J. Klincewicz, A scaled reduced gradient algorithm for network flow problems with convex separable costs, Mathematical Programming Study 15 (1981) 125–147.

[11] M. De Prycker, ATM switching on demand, IEEE Network 6(2) (1992) 25–28.

[12] A. Dutta, Capacity planning of private networks using dcs under multibusy-hour traffic, IEEE Transactions on Communications 42 (1994) 2371–2374.

[13] A. Dutta and J.-I. Lim, A multi-period capacity planning model for backbone computer communication networks, Operations Research 40 (1992) 689–705.

[14] M. Eisenberg, Engineering traffic networks for more than one busy hour, Bell Systems Technical Journal 56 (1977) 1–20.

[15] R. Elbaum and M. Sidi, Topological design of local area networks using genetic algorithms, in: *Proc. of IEEE Conf. on Computer Communications (INFOCOM'95)*, Boston, MA (April 1995) pp. 64–71.

[16] W.B. Elsner, A descent algorithm for the multihour sizing of traffic networks, Bell Systems Technical Journal 56 (1977) 1405–1429.

[17] J.M. Farvolden, W.B. Powell and I.J. Lustig, A primal partitioning solution for the arc-chain formulation of a multi-commodity network flow problem, Operations Research 41 (1993) 669–693.

[18] M. Florian and S. Nguyen, A Method for computing network equilibrium with elastic demand, Transportation Science 8 (1974) 321–332.

[19] L. Fratta, M. Gerla and L. Kleinrock, The flow deviation method: An approach to store-and-forward computer communication network design, Networks 3 (1973) 97–133.

[20] B. Gavish and S. Hantler, An algorithm for optimal route selection in SNA networks, IEEE Transactions on Communications 31 (1983) 1154–1161.

[21] B. Gavish and I. Newman, Capacity and flow assignment in large computer networks, in: *Proc. of IEEE INFOCOM'86* (1986) pp. 275–284.

[22] B. Gavish and I. Neumann, A system for routing and capacity assignment in computer communications networks, IEEE Transactions on Communications 37 (1989) 360–366.
[23] B. Gavish and I. Newman, Routing in a network with unreliable components, IEEE Transactions on Communications 40 (1992) 1249–1258.
[24] M. Gerla, J.A. Suruagy Monteiro and R. Pazos, Topology design and bandwidth allocation in ATM nets, Journal on Selected Areas in Communications 7 (1989) 1253–1262.
[25] A. Girard, *Routing and Dimensioning in Circuit-Switched Networks* (Addison-Wesley, Reading, MA, 1990).
[26] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison-Wesley, Reading, MA, 1989).
[27] C.D. Ha, A generalization of the proximal point algorithm, SIAM Journal of Control and Optimization 28 (1990) 503–512.
[28] M. Held, P. Wolfe and H. Crowder, Validation of subgradient optimization, Mathematical Programming 6 (1974) 62–88.
[29] P. Jog, J.Y. Suh and D. Van Gucht, Parallel genetic algorithms applied to the travelling salesman problem, SIAM Journal on Optimization 1 (1991) 515–529.
[30] J. Hui, M. Gursoy, N. Moayeri and R. Yates, A layered broadband switching architecture with physical or virtual path configurations, IEEE Journal on Selected Areas in Communications 9 (1991) 1416–1425.
[31] S. Lanphongpanich and D. Hearn, Simplical decomposition of the assymetric traffic assignment problem, Transportation Research 18 B (1984) 891–904.
[32] E.L. Lawler, *Combinatorial Optimization: Networks and Matroids* (Holt, Rinehart and Winston, New York, 1976).
[33] D.N. Lee, K.T. Medhi, J. Strand, R. Cox and S. Chen, Solving large telecommunications network loading problems, AT&T Technical Journal 68(3) (1989) 48–56.
[34] F.Y.S. Lin and J.R. Yee, A new multiplier adjustment procedure for the distributed computation of routing assignments in virtual circuit data networks, ORSA Journal on Computing 4 (1992) 250–266.
[35] M. Logothetis and S. Shioda, Centralized virtual path bandwidth allocation scheme for ATM network, IEICE Transactions on Communications E 75-B(10) (1992) 1071–1080.
[36] H. Luss, Operations research and capacity expansion problems: A survey, Operations Research 30 (1982) 907–947.
[37] D. Medhi, A unified approach to network survivability for teletraffic networks: Models, algorithms and analysis, IEEE Transactions on Communications 42 (1994) 534–548.
[38] D. Medhi, Multi-hour, multi-traffic class network design for virtual path-based dynamically reconfigurable wide-area ATM networks, IEEE/ACM Transactions on Networking 3 (1995) 809–818.
[39] D. Medhi, Models for network design, servicing and monitoring of ATM networks based on the virtual path concept, Computer Networks & ISDN Systems 29(3) (1997) 373–386.
[40] D. Medhi and C.D. Ha, Generalized proximal point algorithm for convex optimization, Journal of Optimization Theory and Applications 88(2) (1996) 475–488.
[41] M. Minoux, *Mathematical Programming – Theory and Algorithms* (Wiley, Chichester, 1986).
[42] H. Mühlenbein, Parallel genetic algorithms in combinatorial optimization, in: *Computer Science and Operations Research: New Developments in their Interfaces*, eds. O. Balci, R. Sharda and S. Zenios (Pergamon, 1992) pp. 441–453.
[43] T.M. Ng and D.B. Hoang, Joint optimization of capacity and flow assignment in a packet-switched communications network, IEEE Transactions on Communications 35 (1987) 202–209.
[44] K.E. Nygard and C.-H. Yang, Genetic algorithms for the traveling salesman problem with time windows, in: *Computer Science and Operations Research: New Developments in their Interfaces*, eds. O. Balci, R. Sharda and S. Zenios (Pergamon, 1992) pp. 411–423.

[45] W. Oettli and W. Prager, Optimal and suboptimal capacity allocation in communication networks, Journal of Optimization Theory and Applications 8 (1971) 396–411.
[46] M. Pioro and B. Wallstrom, Multihour optimization of non-hierarchical circuit-switched communication networks with sequential routing, in: *11th Internat. Teletraffic Congress* (1985).
[47] R.T. Rockafellar, Augmented Lagrangian and applications of the proximal point algorithm in convex programming, Mathematics of Operations Research 1 (1976) 97–116.
[48] E. Rolland and H. Pirkul, Heuristic solution procedures for the graph partioning problem, in: *Computer Science and Operations Research: New Developments in their Interfaces*, eds. O. Balci, R. Sharda and S. Zenios (Pergamon, 1992) pp. 475–490.
[49] E. Rosenberg, A nonlinear programming heuristic for computing optimal link capacities in a multihour alternate routing communications network, Operations Research 35 (1987) 354–367.
[50] R. Siebenhaar, Optimized ATM virtual path bandwidth management under fairness constraints, in: *Proc. of IEEE GLOBECOM'94* (December 1994) pp. 924–928.
[51] R.L. Smith, Deferral strategies for a dynamic communication network, Networks 9 (1979) 61–87.
[52] W. Stallings, *ISDN and Broadband ISDN with Frame Relay and ATM*, 3rd ed. (Prentice-Hall, Englewood Cliffs, NJ, 1995).
[53] B. Yaged, Minimum cost routing for dynamic network models, Networks 3 (1973) 193–224.
[54] N. Zadeh, On building minimum cost communications network over time, Networks 4 (1974) 19–34.