A Common Representation of QoS Levels for Resource Allocation in Hybrid Satellite/Terrestrial Networks

Laura Rosati[1], Gianluca Reali[2]

[1]German Aerospace Center (DLR), Institute of Communications and Navigation
P.O. Box 1116, Oberpfaffenhofen, Germany
Phone: +49 8153 282801, Fax: +49 8153 281871, e-mail: laura.rosati@dlr.de
[2]University of Perugia, Department of Electronic and Information Engineering (DIEI)
via G. Duranti 93, 06125 Perugia, Italy
Phone: +39 075 5853651, Fax: +39 075 5853654, e-mail: gianluca.reali@diei.unipg.it

Abstract. The future hybrid satellite/terrestrial networks are challenged to provide different end-to-end Quality of Service (QoS) classes to the users. In particular, it is expected that QoS routing procedures are implemented, in order to decides the best route to the destination and optimize resource distribution through the path. This fact arises the question of a common representation of the QoS levels such that they can be compared and combined. It was shown in the literature how they can be mapped into a single parameter, the so-called virtual delay. We present and investigate the computational complexity of two approaches to distribute the virtual delay over the communication path (i.e., allocating the resources) in order to guarantee a QoS performance requested by the user minimizing the cost of the provided service. The former consists of a joint routing/resource allocation optimization. The latter assumes that the selection of the path is correctly performed and models the resource allocation problem as an equality constrained convex minimization problem.

1 Introduction

ITU-T defines Quality of Service (QoS) as "the collective effect of service performance which determine the degree of satisfaction of a user of the service" [1]. The IETF has proposed QoS architectures to provide guaranteed service level to different applications over *terrestrial* networks. These architectures include Integrated Services (IntServ) [2], Differentiated Services (DiffServ) [3] and MultiProtocol Label Switching (MPLS) [4].

Also the future global *satellite* networks will likely use some on-board switching techniques enabling the provision of service level guarantees. In [5] a QoS framework for satellite IP networks including requirements, objectives and mechanisms is described. In [6] different combinations of buffer management policies to be adopted in satellite systems are presented to guaranty the QoS required by the user.

In this paper we investigate the issue in a *hybrid satellite/terrestrial* environment. The research challenges and technology advances needed to accomplish the integrated format are presented in [7]. The space segment is expected to operate in the future in collaboration with the terrestrial component in order to provide a complementary rather than an alternative service. In particular it is expected that the

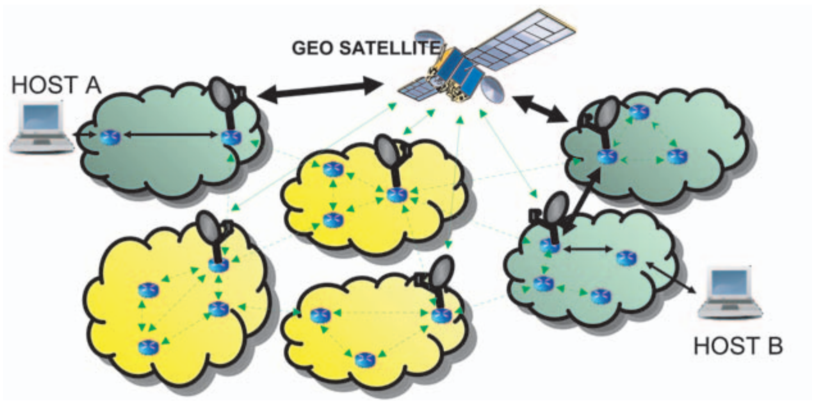

Fig. 1. Hybrid satellite/terrestrial scenario.

satellite will be no longer seen as a component of an alternative routing path but as a part of unique (really integrated) system.

We assume that the system accepts user's requests made in terms of QoS parameters. To achieve a seamless integration, each component of the heterogeneous network is challenged to be "end-to-end QoS-aware". In particular, it is expected that QoS routing procedures are implemented, in order to define for each communication flow the best source-destination path (as depicted in Fig. 1), verify if the route has sufficient resources the QoS requested by the user and optimize resource allocation through the path. In particular, since radio resources are costly and scarcely available and allowance has to be made for housekeeping procedures, the optimization of link layer is of paramount importance.

In this paper we assume that the global network is composed of administratively independent domains, which can be either a terrestrial or a satellite component. Each domain implements admission control and resource allocation functions to provide different levels of IP QoS. In order to accomplish this task, for instance, the satellite can use Bandwidth on Demand (BoD) techniques, allocating an amount of bandwidth to the user on the basis of the QoS requested by the user itself. This approach can be combined with non-uniform *traffic based* bandwidth allocation among the beams constituting the satellite coverage region (for instance beam hopping techniques [10]) in order to enhance the performance of the system in terms of bandwidth efficiency.

The provision of QoS-oriented services in hybrid networks arises the question of a common representation of the QoS levels such that they can be compared and combined. The problem of the mapping of the quality of the network support into a single parameter was faced in [14, 15, 16].

The QoS parameters can be classified as service parameters and network parameters. The service parameters are those that can be defined at call level, and may be negotiated between users and network. Typical service parameters are the transfer delay, the delay jitter, and the packet loss probability. All other parameters that influence the QoS are called network parameters. For instance the network resilience, which characterizes the intrinsic quality of the network, is not specific for individual flows, and is a typical network parameter. The service parameters were used to define the virtual delay d.

An intuitive, although partial, rationale of this model comes from the observation that both delay jitter and loss probability may be traded with delay. Indeed, the delay jitter could be reduced by using a playout buffer at the network egress at the cost of an additional for queuing delay. Hence, a possible model used for describing the actual delay jitter is the equivalent queueing delay. A similar approach may be used for packet loss probability due to buffer overflow. Consequently, a given network service with specific guarantees on delay jitter and packet losses may be modeled as an equivalent service with a given virtual delay, without any delay jitter or losses. The interested reader can find in [14, 16] further details concerning the computation of the virtual delay from the service parameters.

In summary, it is assumed that a virtual end-to-end delay d is computed from service parameters by summing up the actual edge-to-edge delay and the virtual components representing the QoS parameters. A low virtual delay value indicates a good service and vice versa. In principle the sum could be weighted according to whatever criteria, such as customer sensitivity. If we transfer an information unit from a point A to a point B crossing N domains, with guaranteed delays $d_1 \ldots d_N$, then the total end-to-end virtual delay is

$$d = \sum_{i=1}^{N} d_i. \tag{1}$$

When a customer wants to transfer information, it specifies the service quality desired, thus a total virtual end-to-end allowed delay $d_{\max}$ is associated with it. The virtual delays d_i of each crossed domain are chosen such that they satisfy the end-to-end constraints, i.e.,

$$\sum_{i=1}^{N} d_i \leq d_{\max} \tag{2}$$

In fact, we will consider in the paper

$$\sum_{i=1}^{N} d_i = d_{\max} \tag{3}$$

in order to avoid waste of resources. The number of commodity units may be determined by using a function $f(d)$, where d is the virtual delay. Since such function gives the number of commodity units associated with the information transmission, in [17, 18] it was used to define a pricing strategy for guaranteed network services, depending on both the actually used and the reserved network resources. The end-to-end price of the network support for performance guaranteed services is given by the sum of the single tariffs charged by the domains involved in the end-to-end transfer; analytically,

$$F(\underline{d}) = \sum_{i=1}^{N} \xi_i f(d_i) \tag{4}$$

where $F(\underline{d}) : \mathbb{R}_+^N \rightarrow \mathbb{R}_+$ and, for each crossed domain i, d_i is the guaranteed delay; $\xi_i \in \mathbb{R}_+$ is the price of one commodity unit, which might also depend on the network parameters; $f(d_i) : \mathbb{R}_+ \rightarrow \mathbb{R}_+$ is the cost function that associates a measure with the transfer of each information unit, expressed in commodity units. In principle, each domain could select such function arbitrarily. In particular, this selection will reflect the fact that the satellite bandwidth cost is higher than the terrestrial domains bandwidth cost.

Our task is to determine an algorithm which minimizes the cost of the service provided to users. Essentially it is necessary to solve two problems. The former is to find the end-to-end path through independent domains (*routing problem*), the latter is to distribute the virtual delay over the communication path according to the constraint of Eq. 3 (*resource allocation*).

In this paper we investigate two approaches to distribute the virtual delay over the path. The former is a Minimum Price (M-P) routing algorithm [17, 19] which consists of a *joint* routing / resource allocation optimization. In the following it will be referred to as "joint algorithm". It can be seen as a cross-layer technique involving the network and the link layer in the protocol stack. Some approaches present in the literature investigating the Simultaneous Routing and Resource Allocation (SRRA) problem are listed below. In [20] a controller allocates power and schedules the data to be routed over the links in reaction to channel state and queue backlog information. The same topic was investigated in [21] for a scenario constituted of a multibeam satellite down-link which transmit data to ground locations over time-varying channels. The SRRA problem was formulated in [22, 23] as a convex optimization problem over the network flow variables and the communication variables. The aim of this work was extended in [24] to also address transmission scheduling.

The latter algorithm presented in this paper assumes that the selection of the path is correctly performed and faces the resource allocation problem distributing the total allowed virtual delay over the domains involved, such that the total cost for accessing the service over the selected path is minimized. In the following this approach will be referred to as "disjoint algorithm".

The structure of the paper is as follows. In Section 2 we describe the notation, some mathematical definitions and the network model. In Section 3 we recall some known results on M-P routing algorithms and shows the estimation of the computational complexity of the joint optimization approach. In Section 4 we describe our approach to the problem (i.e., the disjoint algorithm). In particular we model the cost of the service as a convex function of the virtual delays, thus the problem becomes a convex optimization one. In Section 5 we present the disjoint optimization complexity analysis. The relevant simulation results are showed in Section 6. In Section 7 we drive the conclusions of the work.

2 Notation and Definitions

- $\mathbb{R}$: the set of real numbers.
- $\mathbb{R}^n$: the set of real n-vectors ($n \times 1$ matrices).
- $\mathbb{R}^{1xn}$: the set of real n-row-vectors ($1 \times n$ matrices).
- $\mathbb{R}^{mxn}$: the set of real $m \times n$ matrices.
- $\mathbb{R}_+$: the set of nonnegative real numbers, i.e., $\mathbb{R}_+ = \{x \in \mathbb{R} | x \geq 0\}$.
- $\|x\|$: norm of $x \in \mathbb{R}^n$.

- $\| x \|_2$: the euclidean norm of $x \in \mathbb{R}^n$, $\| x \| = (x_1^2 + \ldots + x_n^2)^{1/2}$.
- $x \preceq y$: (if x and y are vectors) component-wise inequality: $x_i \leq y_i \forall i$.
- *dom f*: domain of function *f*.
- $f: A \to B$: f is a function on the set $dom\, f \subseteq A$ into the set B.
- $f'(x)$: *first derivative* of a differentiable function $f: \mathbb{R} \to \mathbb{R}$ evaluated at x.
- $f''(x)$: *second derivative* of a twice differentiable function $f: \mathbb{R} \to \mathbb{R}$ evaluated at x.
- $f'''(x)$: *third derivative* of a three times differentiable function $f: \mathbb{R} \to \mathbb{R}$ evaluated at x.
- $\nabla f(x)$: *gradient* of a differentiable function $f: \mathbb{R}^N \to \mathbb{R}$ at x : $(\nabla f(x)_i) = \frac{\partial f}{\partial x_i}$ evaluated at x.
- $\nabla^2 f(x)$: *Hessian* of a twice differentiable function $f: \mathbb{R}^N \to \mathbb{R}$ at x:

  $(\nabla^2 f(x)_i) = \dfrac{\partial^2 f}{\partial x_i \partial x_j}$ evaluated at x.
- A function $f: \mathbb{R}^N \to \mathbb{R}$ is *convex* if *dom f* is a convex set and if $\forall x, y \in dom\, f$ and $\forall \theta \in \mathbb{R}$ with $0 \leq \theta \leq 1$, we have

$$f(\theta x + (1-\theta) y) \leq \theta f(x) + (1-\theta) f(y). \tag{5}$$

- A function $f: \mathbb{R}^N \to \mathbb{R}$ is *strongly convex* on S if there exists an $m_{\mathrm{SC}} > 0$ and an $M_{\mathrm{SC}} > 0$ such that

$$m_{\mathrm{SC}} I \preceq \nabla^2 f(d) \preceq M_{\mathrm{SC}} I \quad \forall d \in S. \tag{6}$$

- A convex function $f: \mathbb{R} \to \mathbb{R}$ is *self-concordant* if $\forall d \in dom\, F$

$$|f'''(d)| \leq f''(d). \tag{7}$$

- A function $f: \mathbb{R}^N \to \mathbb{R}$ is *self-concordant* if it is self-concordant along every line in its domain, i.e., if the function $\tilde{f}(t) = f(d + tv)$ is a self-concordant function of $t \,\forall d \in dom\, f$ and $\forall v$.

2.1 Network Model

We model the topology of a data network by an undirected graph. In this model a collection of n nodes, labeled by $i = 1 \ldots n$, may send, receive, and relay data across m communication links. We label the node by integers $\xi_i \geq 0$, which represent the cost of each commodity unit per time unit.

3 Existing Routing Algorithms and Complexity Analysis

In this Section, Q denotes the number of paths connecting the source and the destination. Since the number of such paths typically grows exponentially with the number n of nodes of the network, we will express Q as $O(a^n)$, where a is a constant. To determine the computational complexity of the joint algorithm, we compute the total number of floating-point operations (flops) to be executed in the worst case, as a function of various problem dimensions, by neglecting all terms, except the dominant ones. A generic algorithm is said to run in $O(f(n))$ time if for some numbers c and n_0, the processing time of the algorithm is at most $cf(n)\ \forall n \geqslant n_0$.

3.1 M-P Routing Algorithm Based on Min-Plus Convolutions

The joint algorithm is based on the use of min-plus convolutions. The general structure of the cost of each path is:

$$g1,i(d_{1i}) = \min_{0 \le d \le d_{1i}} \left[\xi_i f_i(d) + g1,i-1\,(d_{1i} - d)\right] = \xi_1 f_1 * \xi_2 f_2 * \cdots * \xi_i f_i \qquad (8)$$

For detailed information on min plus algebra, the reader should refer to [25, 26]. For the sake of clearness we quote briefly from [17, 19] the steps which constitute the joint algorithm:

– Starting from the source, all the departing inter-domain paths that do not create loops are considered.
– A metric is associated to each path, obtained by computing the min plus convolutions of the cost functions of all domains of the path, computed in the range [0, d_{max}].
– If M_m paths converge towards the same input port of the generic m-th domain, they are compared. Since the maximum allowed delay is d_{max}, for each delay value in the range [0, d_{max}], only the path relevant to the minimum cost function survives, and all other paths are discarded.
– At the destination domain, the values of the cost functions of the paths survived, computed at d_{max}, are compared, and the cost function corresponding to the minimum value is selected.
– Finally, the maximum delay d_{max} is distributed over the selected domains.

3.2 Joint Optimization Complexity Analysis

Since the domain of the cost functions are meaningful only between 0 and d_{max}, this holds also for their convolution. Therefore each convolution may be restricted to the meaningful range. It turns out that the cost of each step is constant, denoted as C_{M-P}.

Fact 1: In the worst case the total cost of the joint algorithm is

$$C_{M-P}\, O\,(a^n n). \qquad (9)$$

Proof: In the worst case, the network is very dense, that is every node is adjacent to every other node, and every path is composed of up to n domains; further, at each domain, we have to compare the relevant cost function, defined in the range [0, d_{max}], with that of the an different paths which could converge towards the domain. In summary, in the worst case we must compute $O(a^n)$ min-plus convolutions. Then, we must compare the values of the cost functions associated with the Q paths. This step requires $Q - 1$ inequalities, i.e., $Q - 1$ flops (a number negligible which does not influence the asymptotical computational complexity of the algorithm). At the end of the algorithm, once we have found the optimum path, we can find the optimal distribution of the virtual delays in few flops (also in this case this number is negligible with respect to the total computational complexity of the algorithm). On the basis of the previous observations, we can estimate that in the worst case the total cost of the joint algorithm is given by Eq. 9.

3.3 Observations

A joint optimization in the terms ξ and *f(d)* characterizes the joint algorithm. Despite this complexity, the relevant algorithms could make sense in operation if the number of involved domains is low (e.g., regional or national). On the contrary, if the number of considered domains is large (world-wide communications), a different solution is necessary. In [17] Authors have also proposed a simpler approach by means of discretization of the domain of the cost functions. This way the possible values of the virtual delay belong to a discrete and finite set, which simplifies the routing algorithm at the detriment of the flexibility in allocating the virtual delay values. Another approach assumes that the selection of the path is correctly performed. Note that the setting of the term ξ will reflect the fact that the price of one commodity unit processed by the satellite is higher than the correspondent price for a terrestrial domain. As a consequence a solution with a low computational complexity may be found by considering the terms ξ_i only, which is the cost of each commodity unit in the *i*-th domain. We stress that in general this approach cannot *guarantee* the optimum solution, but rather a satisfactory solution with a convergence time much lower than the one obtainable by facing the general problem. Thus, the path could be found considering only the term ξ. This type of problem is known in literature as shortest path problem, specifically based on a metric that is the amount of money per commodity unit per time unit ξ. Due to this metric, in this work we will use the term *cheapest* path instead of shortest path. Then the problem becomes an optimization one; we have to find the virtual delays $d_1 \ldots d_N$ (with $\sum_{i=1}^{N} d_i \le d_{\max}$), so as to minimize $F(d)$. It is worth noting that we do not care about the specific technique used within domains to guarantee QoS. We only need an abstract edge-to-edge description, which is the virtual delay and its related cost function.

4 M-P Routing Through Disjoint Optimization

In this Section we investigate the disjoint algorithm. As discussed in Section 3.3, we note that the source-destination paths could be defined on the basis of the commodity price. This way we can pre-select a number of candidate solutions over the same physical path, characterized by the lowest commodity price. After this, the best solution (i.e., virtual delay distribution over the path) is determined. We observe that this approach cannot guarantee the best solution over the network if the cost functions of the domains are very different. A good and satisfactory solution is found any way. On the contrary, if the cost functions are identical, it is trivial to show that the path that has the minimum commodity price include the best solution, found by the subsequent optimization. Between the two extremes, it is expected that if the cost function are similar, even if not identical, the proposed approach can either provide either the best solution or a very good one, very close to the optimum.

Once we have defined the cheapest path between two domains A and B, our task is to find the virtual delays $d_1, \ldots, d_N$ associated with the domains $1, \ldots, N$ of the path so as to minimize the total network cost with QoS guarantees. We present a general approach which can be applied to a general choice of $F(\underline{d})$. We assume that $F(\underline{d})$ is twice continuously differentiable, and strongly convex with constants m_{sc} and M_{sc}.

4.1 Equality Constrained Minimization Problem

Once defined the cheapest path, from Eq. 4 and Eq. 3, the disjoint algorithm can be formulated as follows:

$$\begin{aligned}&\text{minimize } F(\underline{d}) = \sum_{i=1}^{N} \xi_i f(d_i)\\ &\text{subject to } A\underline{d} \;= d_{\max},\end{aligned} \tag{10}$$

where $\underline{d} = (d_1, \ldots, d_N)$ are the optimization variables, $F(\underline{d}){:}\mathbb{R}_+^N \to \mathbb{R}_+$ is convex and $A = (1 \ldots 1)$ with $A \in \mathbb{R}^{1\times N}$. The physical dimension of $\underline{d}$ is the time, thus $\underline{\mathrm{d}} \geq 0$.

We denote $F!$ as the optimal value of this problem, i.e., $F! = \mathit{inf}\{F(\underline{d}) \mid A\underline{d} = d_{\max}\}$. A point $\underline{d}! \in \mathit{dom}\, f$ is optimal for (10) if and only if there is a $\nu! \in \mathbb{R}$ such that

$$\begin{aligned}&A\underline{d}! = d_{\max},\\ &\nabla F(\underline{d}!) + A^T \nu! = 0\end{aligned} \tag{11}$$

hence, solving the equality constrained optimization problem (10) is equivalent to finding a solution of the equations (11), called Karush-Kuhn-Tucker (KKT) equations, which are a set of $N+1$ equations in the $N+1$ variables $\underline{d}!$, $\nu!$. There are several general approaches for equality constrained problems. Below we discuss feasible descent methods.

We assume that a suitable starting point $\underline{d}(0)$ is available. This point must be feasible, i.e., $A\underline{d}(0) = d_{\max}$, and $\underline{d}(0) \in \mathit{dom}\, F$. These methods are called *descent* because all iterates $\underline{d}(k)$ are feasible and $F(\underline{d}(k+1)) < F(\underline{d}(k))$, except when $\underline{d}(k)$ is optimal. The outline of a general feasible descent method for equality constrained minimization is as follows:

Feasible descent method for equality constrained minimization: given a starting point $\underline{\mathrm{d}}(0) \in$ *dom F*, $A\ \underline{d} = \alpha_{\max}$ *repeat*

- *Determine a feasible descent direction* v, $Av = 0$
- *Line search. Choose a step size* $t_{\mathrm{ls}} > 0$
- *Update.* $\underline{d}{:} = \underline{d} + t_{\mathrm{ls}}\, v$

until the stopping criterion is satisfied.

We adopt the same line search for all the descent methods, thus the step which differentiates the different descent methods is the determination of the feasible descent direction v. The *projected gradient method* uses as search direction the Euclidean projection of the negative gradient $-\nabla F$ on the set of feasible directions; *the steepest descent method* uses a step of unit norm that gives the largest decrease in the linear approximation of F; the step used in *the Newton method* is defined as the quantity that must be added to d to solve the problem when the quadratic approximation is used in place of F. The iterates $v(k)$ that converge to an optimal dual variable $v!$, i.e., satisfy

$$\lim_{k \longrightarrow \infty} \nabla F(\underline{d}(k)) + A^T v(k) = 0. \tag{12}$$

The stopping criterion for a feasible descent method generally has the form

$$\left\| \nabla F(\underline{d}(k)) + A^T v(k) \right\| \leq \eta \tag{13}$$

where η is small. This is justified in [27].

4.1.1 Line Search. Different kinds of line search exist [29, 31]. Below we show the results obtainable by using the *backtracking* line search. It depends on two constants α_{ls}, β_{ls} with $0 < \alpha_{\text{ls}} < 0.5$, $0 < \beta_{\text{ls}} < 1$. The step length t_{ls} is chosen to approximately minimize F along the ray $\{d + t_{\text{ls}}\, v \mid t_{\text{ls}} \geq 0\}$:

```
Backtracking line search:
given a descent direction $v$ for $F$ at $\underline{d} \in dom\,F$
$t_{ls} := 1$.
while $(F(\underline{d} + tv) > F(\underline{d}) + \alpha_{ls}\, t_{ls}\, \nabla F(\underline{d})^T v)$
$t_{ls} := \beta_{ls}\, t_{ls}$.
end
```

The line search is called backtracking since it starts with unit step size and then reduces it by the factor β_{ls} until the stopping condition $F(\underline{d} + tv) \leq F(\underline{d}) + \alpha_{\text{ls}}\, t_{\text{ls}}\, \nabla F(\underline{d})^T v$ holds.

4.1.2 Projected Gradient Method and Steepest Descent Method. Any norm $\|\cdot\|$can be bounded in terms of the euclidean norm, i.e., there exists a constant $\gamma \in (0, 1]$ such that $\|\underline{d}\| \geq \gamma \|\underline{d}\|_2$. It can be shown [27] that:

$$F(\underline{d}(k) - F!) \leq \mathrm{c}_{\mathrm{lin}}^{\mathrm{k}} (F(\underline{d}(0) - F!)) \tag{14}$$

where $c_{\text{lin}} = \min\,\{2m_{\text{sc}}\, \alpha_{\text{ls}},\ 2\, \alpha_{\text{ls}}\, \beta_{\text{ls}}\, m_{\text{sc}}/M_{\text{sc}}\} < 1$ for the gradient method, and $c_{\text{lin}} = 1 - m_{\text{sc}}\, \alpha_{\text{ls}}\, \gamma^2 \min\,\{1, \beta_{\text{ls}}\, \gamma^2/M_{\text{sc}}\} < 1$ for the steepest descent method. Thus, in the gradient and in the steepest descent method, $F(\underline{d}(k))$ converges to $F!$ at least as fast as a geometric series with an exponent that depends on the condition number bound $M_{\text{sc}}/m_{\text{sc}}$. In the terminology of iterative methods, the convergence is said at *least linear*.

4.1.3 Newton's Method. We replace the objective with its second order Taylor approximation near $\underline{d}$, to formulate the problem:

$$\text{minimize } \tilde{F}(\underline{d} + \nu) = F(\underline{d}) + \nabla F(\underline{d})^T \nu + \frac{1}{2} \nu^T \nabla^2 F(\underline{d}) \nu \tag{15}$$

$$\text{subject to } A(\underline{d} + \nu) = d_{max} \tag{16}$$

with variable ν. This is a (convex) quadratic minimization problem with equality constraints, and can be solved analytically. The Newton step ν_{nt} is defined as the quantity that must be added to d to solve the problem when the quadratic approximation is used in place of F.

Another alternative is provided by a family of algorithms for unconstrained optimization called quasi-Newton methods [30, 29, 31]. These methods require less computational effort to form the search direction, but since they share some of the strong advantages of Newton methods, such as rapid convergence near $d!$, we will not consider them. The interested reader can find more details about Newton's method in [30, 31]. Among the different feasible descent methods, we have decided to focus our attention on Newton's method for the reasons, experimentally verified, shown in below.

4.1.4 Comparison of the Feasible Descent Methods. Concerning the gradient and the steepest descent methods can be observed:

- The choice of backtracking parameters α_{ls}, β_{ls} has a noticeable but not dramatic effect on convergence.
- The methods often exhibit approximately linear convergence, i.e., the error $F(\underline{d}(k)) - F!$ converges to zero as a geometric series.
- The convergence rate depends greatly on the condition number of the Hessian, that is the ratio of value the largest and the lowest eigenvalues.

About the Newton method we can say that it has several very important advantages over gradient and steepest descent methods:

- It scales well with problem size.
- The good performance of Newton's method is not dependent on the choice of algorithm parameters. In contrast, the choice of the norm for the steepest descent plays a critical role in its performance.
- Its convergence is rapid in general, and quadratic near $\underline{d}!$.
- Once the quadratic convergence phase is reached, a number close to 6 iterations, if the value of η is between 10^{-6} and 10^{-7}), are required to find a very accurate solution.

For the above reasons, in the following we will focus our attention on the Newton's method computational complexity.

5 Disjoint Optimization Complexity Analysis

5.1 Complexity of the Cheapest Path Problems

The disjoint algorithm assumes that the path selection is correctly performed. Routing in Internet is mainly performed by means of two mechanisms: Border Gateway Protocol (BGP) [32] for inter-domain routing and Open Shortest Path First (OSPF) [33] is used for intra-domain routing. In this Section we investigate the computational complexity that the routing process would require if implemented as discussed in Section 3. A cheapest path problem consists in finding a path of minimum cost from a specific source node to another specified sink node, assuming that each link has an associated cost. Hence, in this phase we do not consider any virtual delay, but associate the network connections with the commodity price ξ only, and find the minimum cost path by using known algorithms. The network flow literature [34] typically classifies algorithmic approaches for solving cheapest path problems into two groups: label setting and label correcting. As regards the label setting algorithms we have considered a simple implementation of them, Dijkstra algorithm, then two versions [34] of it: Dial's implementation and R-HEAP implementation. We have also used a special implementation of label correcting algorithm that requires polynomial time. We denote $max\ \{\xi_i, i = 1, \ldots, n\}$ as ξ_{max}. Depending on the values assumed by *n, m*, and ξ_{max} we can select the one which gives the best performance. About the three different versions of Dijkstra algorithm shown in [34], we can say that the original $O(n^2)$ one has the optimal running time for fully dense networks (with at least n^2 arcs). A potential disadvantage of this Dial's scheme, as compared to the original one, is that ξ_{max} may be very large, thus requiring large storage and increased computational time. The R-heap implementation runs in $O(m + n \log \xi_{max})$ time units. Using more sophisticated data structures, it is possible to reduce this bound to $O(m + n\sqrt{\log n})$, which is a linear time algorithm for all but the sparsest classes of shortest path problems. Label setting has the most attractive worst-case performance but practical experience has shown that label correcting is fairly more efficient. In the following, we will express the computational complexity of a cheapest path problem as

$$O\left(S(n, m, \xi_{\max})\right). \tag{17}$$

5.2 Complexity of the Newton's Method

The Newton's method for solving an equality constrained minimization problem is constituted of the 4 main steps described in 4.1. We assume that the cost of the first step is negligible. The other steps are repeated until the stopping criterion is satisfied, thus to obtain the total computational time we must multiply the running time of these steps by the number of iterations. If $F(\underline{d})$ is strongly convex and the value of η is between 10^{-6} and 10^{-7}, the number of iterations is upper bounded by [27]:

$$6 + (\alpha_{\text{ls}}\beta_{\text{ls}} \min\{1, 3(1 - 2\alpha_{\text{ls}})\})\,\frac{m_{\text{sc}}^3}{M_{\text{sc}}^2 L_1}\,(F(\underline{d}(0)) - F!) \tag{18}$$

If $F(\underline{d})$ is self-concordant, for the same η values the number of iterations is upper bounded by [27]:

$$\frac{20 - 8\alpha_{\text{ls}}}{\alpha_{\text{ls}}\beta_{\text{ls}}(1 - 2\alpha_{\text{ls}})^2}\,(F(\underline{d}(0)) - F!) + 6 \tag{19}$$

This expression depends only on the line search parameters α_{ls} and β_{ls}. If, for example, we take $\alpha_{\text{ls}} = 0.1$ and $\beta_{\text{ls}} = 0.9$ the previous expression becomes $334(F(\underline{d}(0)) - F!) + 6$. The entire line search can be carried out at an effort comparable to simply evaluating F, thus the step which requires the largest computational time is the computation of the Newton step.

Fact 2: The number of flops required by the Newton step for our problem may be expressed in the form:

$$N_{W-f} = t_{\mathrm{H}} + t_{\mathrm{g}} + 5N. \tag{20}$$

where t_{H} and t_{g} are the times necessary to calculate $H = \nabla^2 F(\underline{d})$ and $g = \nabla F(\underline{d})$.

Proof: In this Section we describe methods that can be used to compute the Newton step, i.e., to solve the KKT system

$$\begin{bmatrix} H & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} -g \\ 0 \end{bmatrix} \tag{21}$$

where $H = \nabla^2 F(\underline{d})$ and $g = \nabla F(\underline{d})$, for v (and w). A simple straightforward approach is to solve the KKT system which is a set of $N + 1$ linear equations in $N + 1$ variables. The KKT matrix is symmetric and positive definite. By eliminating ν from the KKT system and solving for ω we obtain the reduced equations:

$$w = -(AH^{-1}A^T)^{-1}AH^{-1}g \tag{22}$$

$$\nu = H^{-1}(-A^T w - g) \tag{23}$$

which give us an alternate method for computing ν and w, constituted of the following steps:

1) Form $H^{-1}A^T \in \mathbb{R}^{N\times 1}$ and $H^{-1}g \in \mathbb{R}^{N\times 1}$. We have:

$$H = diag(\nabla^2 F(\underline{d})_{11} \ldots \nabla^2 F(\underline{d})_{NN}) \tag{24}$$

   and

$$H^{-1} = diag\left((\nabla^2 F(\underline{d})_{11})^{-1} \ldots (\nabla^2 F(\underline{d})_{NN})^{-1}\right). \tag{25}$$

   The term of the i-row of $H^{-1} A^T$ is $(\nabla^2 F(\underline{d})_{ii})^{-1}$. The term of the i-row of $H^{-1} g$ is: $(\nabla^2 F(\underline{d})_{ii})^{-1} \nabla F(\underline{d})_i$.

2) Form $S = -AH^{-1} A^T$ We note that $S \in \mathbb{R}$. We have:

$$S = -\sum_{i=1}^{N} (\nabla^2 F(\underline{d})_{ii})^{-1} \tag{26}$$

3) Form $w = S^{-1} A (H^{-1} g)$. We note that w $\in \mathbb{R}$. We have:

$$w = S^{-1} \sum_{i=1}^{N} \left((\nabla^2 F(\underline{d})_{ii})^{-1} \nabla F(\underline{d})_i\right). \tag{27}$$

4) Form $v = H^{-1}(-A^T w - g)$. We note that $v \in \mathbb{R}^{N\times 1}$. The term of the i-row of v is:

$$\left(w - \nabla F(\underline{d})_i)\right)(\nabla^2 F(\underline{d})_{ii}\Big)^{-1} \tag{28}$$

Now we estimate the total computational complexity associated with each step:

– We define t_{H} and t_{g} as the time necessary to calculate H and g, respectively. To compute H^{-1} we need N divisions, thus N flops. We need N flops also to calculate H^{-1} g, while H^{-1} A^T does not need any further flop since it is constituted of the elements of H^{-1}.

- To form S we have to sum up the N terms $\neq 0$ of H^{-1}, thus we need $N - 1$ flops.
- To form w, we have to sum up the N terms $\neq 0$ of $H^{-1}\, g$ and then multiply for S^{-1}, thus we need $N + 1$ flops.
- To form v, we have to calculate N–1 sums and N divisions, thus we need $2N$–1 flops.

Thus, the number of flops of the step is

$$N_{W-f} = t_{\mathrm{H}} + t_{\mathrm{g}} + 5N. \tag{29}$$

In summary, if the cost function is self-concordant, $\alpha_{\mathrm{ls}} = 0.1$, and $\beta_{\mathrm{ls}} = 0.9$, the worst-case complexity can be expressed as:

$$(334\,(F(\underline{d}(0) - F!)) + 6)\,(t_{\mathrm{H}} + t_{\mathrm{g}} + 5N). \tag{30}$$

If we take into account also the routing algorithm, then from Eq. 17 we have that the total computational complexity of the disjoint algorithm is:

$$(334\,(F(\underline{d}(0) - F!)) + 6)\,(t_{\mathrm{H}} + t_{\mathrm{g}} + 5N) + O(S(n, m, \xi_{\max})), \tag{31}$$

that is polynomially bounded.

6 Simulation Results

Now we make some consideration about the choice of $f(d)$. This is a cost function, thus it must be chosen such that $f(d) \geq 0 \forall d \in dom\ f$; it is meaningful if it is monotonic nondecreasing since the cost has to decrease with the value of the virtual delay ($f'(d) \leq 0$); moreover, in order to use convex optimization algorithms, it is desirable for it to be convex ($f''(d) \geq 0$), in particular strongly convex or self-concordant. Since these properties are preserved by sums, we can conclude that the same properties are valid for Eq. 4.

The use of strongly convex and self-concordant functions is important since for these classes of functions we can determine an upper bound to the number of iterations of the algorithm. Any case, we have experimentally verified that this number is typically pretty much lower than this bound. In addition, the feasible methods described in Section 4.1 can be applied also to functions which are not strongly convex or self-concordant with good results from the point of view of the processing time. We considered the following function:

$$f(d) = e^{-xd + k} \tag{32}$$

with $x \in \mathbb{R}_+ - \{0\}, k \in \mathbb{R}$ and $dom\ f = \mathbb{R}$, as cost function associated with the satellite domain. The physical dimension of $\underline{d}$ is the time, thus $\underline{d} \succeq 0$. We have $f(d) > 0$, $f'(d) < 0$ and $f''(d) > 0 \forall d \in \mathbb{R}_+$.

We considered the following function:

$$f(d) = -\log\left(\frac{d}{d_{\max}}\right) + \frac{m}{2}d^2 \tag{33}$$

with m $\in \mathbb{R}_+ - \{0\}$ and $dom\ f = \mathbb{R}_+ - \{0\}$, as cost function associated with the terrestrial domains.

Since $d < d_{\max}$, we have $f(d) > 0 \forall d \in dom\ f$; if $d < \sqrt{\frac{1}{m}}$ then $f'(\mathrm{d}) < 0$. We can note that $f(d)$ is strongly convex $\forall d \in dom\ f$. As a consequence, if we assume that each path comprises at least one terrestrial domain, also $F(d)$ is strongly convex, thus efficient algorithms can be computed to minimize it without verifying if it is self-concordant.

We considered a network constituted of one satellite and three groups of terrestrial domains. The first group, constituted of 10 domains, implements the cost function (33) with $m = 1$; the second one, constituted of 20 domains the same cost function with $m = 3$, the third one, constituted of 30 domains adopts $m = 5$. The satellite implements the cost function (32) with $x = 2$ and $k = 1$. The values of ξ are: 10 for the satellite, 1 for the first group of terrestrial domains, 2 for the second and 3 for the third one.

We implemented by using MATLAB the three feasible optimization methods described in Section 4 setting η to 10^{-6} and assuming as starting feasible point $\underline{d}(0) = \{1, \ldots, 1\}$ (the sum of the virtual delays is constrained to be 61). We used the backtracking line search setting $\alpha_{ls} = 0.1$ and $\beta_{ls} = 0.9$. In Table 1 the main simulation parameters are listed.

Table 1. Simulation parameters.

	x	k	m	ξ	$d_i(0)$
Satellite	2	1	/	10	1
Terrestrial Domains (Group 1)	/	/	2	1	1
Terrestrial Domains (Group 2)	/	/	3	2	1
Terrestrial Domains (Group 3)	/	/	5	3	1

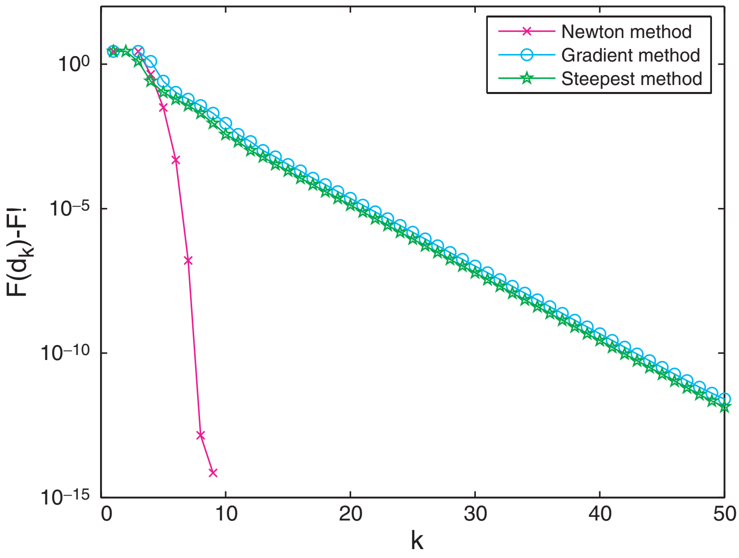

Fig. 2. Error $F(d_k) - F!$ versus iteration number.

In Fig. 2 we plotted, for each algorithm iteration k, the difference between the value of the cost function F evaluated at $d^{(k)}$ and the optimal value $F!$. It can be appreciated that the gradient method and the steepest descent method exhibit approximately linear convergence, i.e., the error $F(\underline{d}(k)) - F!$ converges to zero as a geometric series. In general the convergence of the Newton method is more rapid, and quadratic near $F!$.

7 Conclusions

In this paper we have faced the M-P routing problem, which consists in finding the cheapest path among different independent domains, which charge users for IP network services with guaranteed QoS performance. We have investigated two solutions. The former is based on min-plus convolutions and involves the link and the network layer of the protocol stack. We showed that the computational complexity due to this approach is *exponentially* bounded. The latter formulates the M-P problem as an equality constrained convex problem which can be solved by means of a feasible descent method. We showed the definitions, the mathematical aspects and we focused our attention on the characteristics of the convergence of 3 types of feasible descent methods. We showed that the computational complexity due to this approach is *polynomially* bounded.

Acknowledgment

This work has been partially funded by the European Community under the 6-th Framework Programme IST Networks of Excellence "SatNEx" (contract No. 507052) and "SatNEx II" (contract No. 027393).

References

[1] "Communications quality of service: a framework and definitions," ITU-T Recommendation SG-12 G.1000, 2001.
[2] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: An Overview," IETF RFC 1633, 1994.
[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for Differentiated Services," IETF RFC 2475, 1998.
[4] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," IETF RFC 3031, Jan 2001.
[5] S. Kota and M. Marchese, "Quality of service for satellite IP networks: a survey," *International Journal of satellite communications and networking*, vol. 21, pp. 303–349, 2003.
[6] N. Courville, "QoS-oriented traffic management in multimedia satellite systems," *International Journal of satellite communications and networking*, vol. 21, pp. 367–399, 2003.
[7] F. Daoud, "Hybrid satellite/terrestrial networks integration," *Computer Networks*, vol. 34, pp. 781–797, 2000.
[8] X. Maufroid, R. Rinaldo, and R. C. Garcia, "Analysis of Beam Hopping Techniques in Future Multi-Beam Broadband Satellite Networks," in *Proceedings 23rd International*

Communication Satellite Systems Conference (ICSSC) and 11th Ka and Broadband Communications Conference, Rome, Italy, 2005.

[9] D. K. Okello and M. Kaplan, "Adaptive Beam Allocation for Multimedia Ka-band Satellite Networks," in *Proceedings 59th Vehicular Technology Conference (VTC)*, Milan, Italy, 2004.

[10] L. Rosati and G. Reali, "On Traffic-demand Based Multi-Beam Bandwidth Allocation in Future Satellite Networks Using Beam-Hopping Techniques," in *Proceedings of Advanced Satellite Mobile Systems (ASMS)*, Herrsching, Germany, 2006.

[11] F. Chiti, R. Fantacci, T. Pecorella, L. Giacomelli, M. Poggesi, and F. Poggianti, "A resource allocation scheme based on traffic prediction for DVB-RCS systems," in *Proceedings 59th Vehicular Technology Conference (VTC)*, Milan, Italy, 2004.

[12] F. D. Priscoli and A. Pietrabissa, "Load-adaptive bandwidth-on-demand protocol for satellite networks," in *Proc. of the 41st IEEE Conf. on Decision and control*, Las Vegas, USA, 2002, pp. 4066–4071.

[13] L. Chisci, R. Fantacci, F. Francioli, and T. Pecorella, "Multi-terminal dynamic bandwidth allocation in GEO Satellite Networks," in *Proceedings 59th Vehicular Technology Conference (VTC)*, Milan, Italy, 2004.

[14] D. Di Sorte, M. Femminella, and G. Reali, "A Novel Approach to Charge for IP Services with QoS support," *Journal of Network and Systems Management*, vol. 8, no. 11, 2004.

[15] D. Di Sorte, M. Femminella, and G. Reali, "A QoS Index for IP Services to Effectively Support Usage-based Charging," *IEEE Communications Letters*, vol. 8, no. 11, 2004.

[16] N. Blefari-Melazzi, D. Di Sorte, M. Femminella, and G. Reali, "Theoretical analysis of a virtual delay based tariff model," in *IEEE International Conference on Communications (ICC)*, Anchorage, USA, 2003.

[17] N. Blefari-Melazzi, D. Di Sorte, and G. Reali, "Inter-domain Routing Algorithms that Maximize Users. Benefit in an Internet Business Model," in *Proceedings of IEEE Globecom 2002*, Taipei, Taiwan, 2002.

[18] D. Di Sorte, M. Feminella, G. Reali, and S. Zeisberg, "Network Service Provisioning in UWB Open Mobile Access Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 20, no. 9, 2002.

[19] D. Di Sorte and G. Reali, "Minimum Price Inter-Domain Routing Algorithm," *IEEE Communications Letters*, vol. 6, no. 4, 2002.

[20] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic Power Allocation and Routing for Time Varying Wireless Networks," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, 2003.

[21] M. J. Neely, E. Modiano, and C. E. Rohrs, "Power Allocation and Routing in Multibeam Satellites with Time-Varying Channels," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, 2003.

[22] L. Xiao, M. Johansson, and S. Boyd, "Simultaneous Routing and Resource Allocation via Dual Decomposition," *IEEE Transactions on Communications*, vol. 52, no. 7, pp. 1136–1144, 2004.

[23] M. Johansson, L. Xiao, and S. Boyd, "Simultaneous Routing and Resource Allocation in CDMA wireless data networks," in *IEEE International Conference on Communications*, Anchorage, Alaska, 2003.

[24] M. Johansson and L. Xiao, "Scheduling, Routing and Power Allocation for Fairness in Wireless Networks," in *Proceedings 59th Vehicular Technology Conference (VTC)*, Milan, Italy, 2004.

[25] F. Baccelli, G. Cohen, G. J. Olsder, and J. P. Quadrat, *Synchronization and linearity, an algebra for discrete event systems*. New York: Wiley, 1992.

[26] C. S. Chang, "Deterministic traffic specification via projections under the min-plus algebra," in *Proceedings of IEEE INFOCOM*, New York, USA, 1999.

[27] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Stanford University: EE 364 Course Reader, 2001.
[28] J. Jahn, *Introduction to the Theory of Nonlinear Optimization*. Springer, 1996.
[29] J. Hiriart-Urruty and C. Lemarechal, *Convex analysis and Minimization algorithms*. Springer-Verlag, 1993.
[30] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear programming. Theory and algorithms. Second edition*. Wiley, 1993.
[31] D. P. Bertsekas, *Nonlinear Programming, second edition*. Athena Scientific, 1995.
[32] Y. Rekhter and T. Li, “A border gateway protocol 4 (BGP-4).” RFC 1771.
[33] J. Moy, “OSPF version 2.” RFC 2328.
[34] R. K. Ahujaa, J. L. Magnanti, and J. B. Orlin, *Network flows*. Prentice Hall, 1993.