Robust routing paradigms exploiting path diversity: modeling, analysis and design

Luca Muscariello¹, Bruno Nardelli ¹,², Diego Perino¹
¹ France Télécom R&D - Orange Labs Paris, France, ² Università di Trento, Italy

Abstract—In this paper we consider feedback-based routing exploiting path diversity that jointly uses best path selection and flow control for optimality and stability. The contributions of the present paper are manifold. We show numerical evidence that routing performance are insensitive to rate coordination along available routes in practical scenarios. Route uncoordination arises in case of in-network per flow rate sharing imposed by link schedulers and is known to be suboptimal. However, we show the gap is squeezed out on medium sized networks with different kind of realistic traffic demands. Furthermore, we design a distributed routing algorithm whose performance are evaluated and compared with TRUMP and TEXCP, two recently proposed multi-path routing algorithms. On a US-like backbone network, with and without in-network scheduling our algorithm proves to work as better as the other two while being much simpler in a real protocol implementation. Modeling and analysis of such algorithms is performed through fluid models based on ordinary differential equations. We finalize our design into a real protocol evaluated in an experimental network.

I. INTRODUCTION

Robust routing has many important applications in today’s networks, the most relevant being intra-domain traffic engineering (TE) and inter-AS path selection under the same ISP. In the future, several key applications for an ISP will also require robust routing. Some examples are service overlays and virtual networks that are rapidly gaining popularity thanks to their ease to build and deploy new services.

In the context of intra-domain TE, traditional issues are due to frequent line-card failures, large flow reroutes due to inconsistent routes’ ranking coming from different routing protocols, exposure to highly variable, or unknown, traffic matrices. An additional challenge comes from access line upgrades that will attain, in the near future, capacities of 100Mbps. This will tempt every customer to run video applications (namely P2P-TV) or massive gaming (Internet video game tournaments).

This upgrade process has just started and will exacerbate all the cited performance problems once such access lines will be dominant. Core network saturation is the chronicle of a death foretold that suggests deployments of robust overload control mechanisms.

Up to now, service overlays and virtual networks have ignored optimal resources allocation whilst focusing more on service separation and configuration. The potential network instability induced by the interaction among multiple overlays and virtual networks, with the consequent unpredictable resources allocation, might diminish the initial enthusiasm on these technologies.

In this paper we consider two kinds of network architectures where, respectively, bandwidth sharing is realized by flow control or by in-network link scheduling (fair queuing schedulers). Fair queuing (FQ) has many known desirable effects on resource allocation ranging from fast rate convergence to the insensitivity to a common fairness criterion in transport protocols. This allows new applications to be easily deployed with no need to handle fairness issues. However FQ imposes the fair rate to each flow traversing a link and preventing transmitters from choosing a different rate share.

The contribution of the present paper is threefold. First, we focus on the classic optimization problem of joint routing and flow control and solve it for a set of cases, the most significant being presented in detail. As a result, we find that the overall performance of architectures deploying multi-path routing in medium sized network, with a large number of demands, do not depend on the use of fair queueing. Second, we design a new network routing protocol, MIRTO, from some key observations inferred from optimization. Our routing scheme, that exploits path diversity, is nearly optimal, is adaptive and does not assume demands have infinite backlog. An implementation of the protocol has been made in a service overlay that can be run in common or dedicated hardware and middleboxes. Third, we introduce an analytical model for the comparison of different routing schemes based on fluid ordinary differential equations (ODEs). MIRTO and two other recent algorithms, TEXCP [15] and TRUMP [10], are modeled and compared on the Abilene network topology with FIFO and FQ scheduling. To the best of our knowledge, this is the first time a comparison of the aforementioned routing algorithms is performed via fluid models.

The rest of the paper is organized as follows. Section II includes the problem formulation and related works. In section III we solve a toy case plus a realistic large set of problems. In section IV we design and implement MIRTO in a real platform and in V we model and compare to TRUMP and TEXCP that are modelled as well. In section VI we conclude the paper.

II. PROBLEM FORMULATION AND RELATED WORK

Optimized multi-path routing is an old problem that spurred significant research around it. The first formulation dates back to the work of Gallager on minimum cost routing of datagram’s flow, along multiple routes [5]. The mathematical formulation is that of a multi-commodity flow problem, with convex objectives and linear constraints. If the network has enough capacity, traffic demands are optimally routed, other-
wise the problem is not feasible. This work evolved towards the joint routing and flow control framework, formalized by Golestani and Gallager in late seventies (see [6], [7]), that handles network rate limitations, using both path costs and flow rate adaptation. Kelly tackles the problem under a different perspective ([16], [17]), by introducing the concept of user’s utility. The focus is on rate control and fairness through the study of stability of differential equations in presence of network delays. A thorough description of recent approaches necessitates a formal definition of the problem.

A. Notation

The network topology is modelled by a connected graph $G = (\mathcal{N}, \mathcal{L})$ given as a set of nodes and links. Be $[a_{ij}]$ the adjacency matrix, $a_{ij} = 1$ if there exists a directional link between $i$ and $j$ and $a_{ij} = 0$ otherwise.

The network carries traffic generated by a set of demands $\mathcal{D}$, each characterized by triple $(s^d, e^d, p^d)$, with $s \in \mathcal{S}$, $e \in \mathcal{E}$, source and destination nodes, $\mathcal{S}, \mathcal{E} \subseteq \mathcal{N}$, and $p \in \mathbb{R}^+$ the exogenous peak rate. This latter is the rate the flow would attain if the network had infinite resources. In the model a network flow $d, d \in \mathcal{D}$, gets a share $x_{ij}^d$ of the capacity $C_{ij}$ at each link with $0 \leq x_{ij}^d \leq \min(C_{ij}, p^d)$.

The network flow can be split among different paths that are made available by a network protocol at an ingress node. We make no modeling assumptions on whether paths are disjoint, however the ability to create more path diversity helps design highly robust network routing protocols.

B. User utility and network cost

Utility of a flow $d$, $U^d$ (assumed strictly convex) and network cost $V$ (assumed strictly concave) are two conflicting objectives in a mathematical formulation. $U^d$ is a function of the total rate $y^d$, attained by a flow, split among different paths, and is supposed to be an alpha fair function [7], [23].

The cost $V$ is a function of the link load. Authors in [9], [10], [15] have used the cost function to take into account TE objectives usually representing link load minimization, i.e. minimizing expenditures for line-card upgrades. [8], [13], [18], [19], [26], [28], [29] have used the network cost as penalty function in lieu of capacity constraints in the optimization framework. In our work we interpret network cost $V$ as to select minimum cost routes (path ranking function).

The common mathematical arc-node formulation is as follows :

$$\begin{align*}
\text{maximize} & \quad \sum_{d \in \mathcal{D}} U^d(y^d) - \sum_{i,j \in \mathcal{N}} V\left(\sum_{d \in \mathcal{D}} x_{ij}^d/C_{ij}\right) \\
\text{subject to} & \quad \sum_{k \in \mathcal{N}} a_{ik} x_{ki}^d - \sum_{j \in \mathcal{N}} a_{ij} x_{ij}^d = \phi_i(y^d) \quad \forall i \in \mathcal{N}, d \in \mathcal{D} \\
& \quad \sum_{d \in \mathcal{D}} x_{ij}^d \leq C_{ij} \quad \forall i, j \in \mathcal{L}, d \in \mathcal{D} \\
& \quad y^d \leq p^d \quad \forall d \in \mathcal{D},
\end{align*}$$

where $\phi_i(y^d)$ is the flow in each node $i$ that is zero in every forwarding node, equal to $y^d$ or $-y^d$ in a source or destination node respectively. In the path-demand formulation, the flow rate $y^d$ of a given demand $d$ can be rewritten as the sum of the rates over all paths $p^d$, i.e. $y^d = \sum_{i \in p^d} y_i^d$.

In such a formulation a user coordinates sending rates over paths jointly, aiming at maximizing its own utility. Path coordination is broken when utility is evaluated as the sum of utilities among all paths i.e. $\sum_{i \in p^d} U(y_i^d)$, or because the network might share bandwidth at a link base as in presence of fair queuing algorithms. Uncoordination might also be seen as a relaxation of the objective as the following holds $U(\sum_{p \in \mathcal{P}^d} y_i^d) \geq \sum_{i \in p^d} U(y_i^d)$. In [19], it is has been shown that in a triangular network with uniform demands, without rate limitations, uncoordination could cost a waste of 30%.

Recent work focus on optimal (or nearly-optimal) distributed algorithms and stability in presence of delays while neglecting flow rate limitation. We reintroduce this constraint, already present in the seminal work of Golestani, as it has large impact on optimal routing.

III. OPTIMAL JOINT ROUTING AND FLOW CONTROL

In this section we start by comparing coordinated and uncoordinated multipath routing using optimization. We begin by considering a toy case. Take a network composed of two nodes $A$ and $B$, connected by two links with capacity $C_1$ and $C_2$. A single demand with peak rate $p$ flows from $A$ to $B$. Call $x_1, x_2$, the rate on path one and two respectively. We solve (1) with the following (coordinated) utility and cost,

$$U(y) = -1/y; V(x_1) + V(x_2) = x_1/(C_1-x_1) + x_2/(C_2-x_2)$$

subject to $x_1 + x_2 = y, y \leq p.$ The solution can be found using the Lagrangian multipliers, $L(x_1, x_2, \lambda) = -1/(x_1 + x_2) + x_1/(C_1-x_1)^2 + x_2/(C_2-x_2)^2 + \lambda(x_1 + x_2 - p)$.

Working this out by derivation and assuming $C_1 > C_2$ with no loss of generality we obtain the following solution when

$$C_1 - \sqrt{C_1 C_2} \leq \frac{C_1 + C_2}{1 + \sqrt{C_1 + C_2}}.$$
\((x_1, x_2) = (p \land C_1 - \sqrt{C_1 C_2}, 0)\) and only the dominant path is used. Instead, under the (uncoordinated) utility definition

\[
U(x_1) + U(x_2) = -1/x_1 - 1/x_2
\]

generalize this example to the case of \(n\) paths. Suppose \(C_1 > C_2 > \cdots > C_n\). For small peak rates \(p\), \(\partial U = \partial V_1 < \partial V_2 \leq \cdots \leq \partial V_n\) and only the first path is used. As \(p\) increases up to the point \(\partial U = \partial V_1 = \partial V_2 < \cdots \leq \partial V_n\) the first two are used and so on for larger \(p\) until all paths are used having the same derivative costs.

**Remark 1:** The best path should be used first and filled since path usage attains a certain threshold. If some demand remains unallocated the flow starts filling the second best path and so on.

In the following, we numerically solve (1) for large problems.

**A. Numerical evaluations**

Let us assume max-min as fairness criterion and a linear path cost \(V\). In this case the solution of large-sized problems is possible thanks to the existence of an iterative linear programming (LP) formulations as reported in appendix A (see also [14] and [24]).

We consider the Abilene network topology [1], shown in fig. 2 with \(N = 11\) nodes and link capacities distributed according to a Gaussian law with mean \(\overline{C}\) (between 5 and 100Mps) and standard deviation \(\overline{C}/10\). Two traffic matrix distributions are considered: uniform and hot-spot. Respectively, in the first case demands are established between all couple of nodes, in the second case three nodes transmit to the same destination. The distribution of peak rates is taken as Log-Normal with parameters \(\mu = 16.6\) and \(\sigma = 1.04\) (see [25], [30]). According to the aforementioned set-up we simulate a traffic matrix which is used as input to the LP described in appendix A and solved using the MATLAB optimization toolbox. For every scenario, we evaluate the satisfaction of each demand as the ratio between its attained rate and its exogenous peak rate. We do not account for demands with no rate limitation for which satisfaction would not be defined, however some flows have very high rates. This performance parameter permits to link optimal routing and flow control to the distribution of flow rates. The problems count 110 demands and output data are averaged over multiple runs. Fig. 3 and fig. 4 report three performance measures: average satisfaction as a function of link capacities (plot a), demand satisfaction per flow (plot b) and flow throughput and network usage as a function of link capacities (plot c). By the comparison of figures 3.a/c and fig. 4.a/c we can write the following remark.

**Remark 2:** Optimal routing in presence of uniform traffic matrices does not make use of secondary path. Routing is essentially shortest path. In case of hot-spot demands, even if routing starts using secondary paths, coordinated and uncoordinated routing coincide.

Looking at figures 3.b and fig. 4.b we state the following.

**Remark 3:** Satisfaction decreases as the peak rate increases since shortest paths are more likely to be allocated to rate limited demands whereas high rate flows are constrained to use also secondary (congested) paths.

We have made the same evaluation aggregating flows with respect to the origin-destination in the network (called OD pairs in TE problems) so that multi-path routing is made on large aggregates of flows with different characteristics. The advantage is that the architecture is far more scalable but neglecting the performance of each single flow. We conclude with the following remark:

**Remark 4:** As long as the access rates’ distribution within a traffic aggregate is homogeneous among the different OD pairs, performance is insensitive to aggregation. On the contrary, if the distribution changes among OD pairs, performance of single flows within traffic aggregates are better for high rate flows to the detriment of low rate ones.

**IV. MIRTO, THE NETWORK PROTOCOL**

Multipath iterative routing traffic optimizer, or just MIRTO, is run on access points (ingress nodes) to route traffic demands toward the egress of the network, traversing a number of forwarding nodes. Every node implements signalization functionalities to maintain up to date a series of structural

---

**(Fig. 1. Diagram of the optimal allocation in case of coordinated and uncoordinated multipath: \(x_1, x_2\) rate of flow 1 and 2 respectively.)**

**(Fig. 2. Abilene Network Topology.)**
Fig. 3. Uniform traffic matrix. Average satisfaction for varying link capacities (a) or demand id sort by peak rate (b), and network usage/thruput for varying link capacities (c)

Fig. 4. Hot-spot traffic matrix. Average satisfaction for varying link capacities (a) or demand id sort by peak rate (b), and network usage/thruput for varying link capacities (c)

network informations, as paths state and ranking that are fed back to the access and exit nodes. Core nodes are responsible for the maintenance of routing tables and forwarding paths. They might also implement more complex functionalities to support basic or enhanced explicit congestion notification to the sources or per-flow bandwidth allocation through FQ. Transmitters, on their own, decide how to exploit available paths on the base of path ranking and available bandwidth.

A. Path selection

Following remark 1, MIRTO starts to fill the best ranked path first, until either the network share is larger than its backlog or the network share, being smaller than the requested rate, does not vary "significantly".

In the first case, it means that there is no need to look for bandwidth elsewhere as satisfaction has attained the maximum. In the second case, MIRTO starts to seek resources from the second best ranked path. This procedure iterates over the other paths and stops before probing them all if the demand gets satisfied before, otherwise it continues up to the point it is bottlenecked by network congestion.

B. Probing network resources

The above described procedure implies that all MIRTO transmitters implement a common per path rate fairness criterion. MIRTO could borrow one flow controller from those available and deeply studied in the field of window flow control, e.g. additive/multiplicative increase, multiplicative decrease (AIMD/MIMD). The particular chosen fairness criterion is of little importance in our scope, while it is fundamental to pick the same for all MIRTO agents. The minimum requirement is than that egress nodes ACK or NACK data reception as to signal congestion. Enhanced congestion notifications are possible as ECN or RE-ECN (see [3]).

C. Preventing non optimal allocations

In order to guarantee MIRTO to find optimal or nearly optimal allocations we must avoid that the probing procedure is trapped in non optimal yet stable states: local optima. It might happen that the attained rate along a given path oscillates around the mean. These oscillations are naturally occur due to flow control and depend on the interaction between sources and nodes.

A well understood case is TCP in presence of FIFO queuing with RED, drop tail, fair dropping or ECN. Oscillations depend on the number of flows in progress, on node scheduling, on round trip times (RTT), on buffer sizing and maybe on other unpredictable causes that we ignore in more complex traffic scenarios.

As a matter of fact, it is unwise relying on oscillations. They might also be seen as deficiencies of the flow control protocol and can be removed with more efficient solutions to obtain fast rate convergence. MIRTO estimates the attained rate using a moving average that filters out high frequencies to detect if available bandwidth is truly varying. The cutoff scale of the
filter should be chosen much larger than the round trip time over the path.

We say that a path \( i \) is in *steady state* with respect to a source \( d \) if the mean sending rate of flow \( d \) along path \( i \) varies with respect to its absolute value less than a few percents. Similarly, path \( i \) is said *congested* as long it receives NACKS (or negative congestion notifications). To summarize, the protocol behavior is reported in the algorithms 1 and 2.

![Algorithm 1 MIRTO: ACK reception](image)

**Algorithm 1** MIRTO: ACK reception

- ACK from path \( i \) to source \( d \)
- if isSteady( \( y^d_i \) ) AND isBestSteadyPath( \( y^d_i \) ) then
  - Increase( \( y^d_i \) )
- else if isNotSteady( \( y^d_i \) ) AND isBestNonSteadyPath( \( y^d_i \) ) then
  - Increase( \( y^d_i \) )
- end if

![Algorithm 2 MIRTO: NACK reception](image)

**Algorithm 2** MIRTO: NACK reception

- NACK from path \( i \) to source \( d \)
- DecreaseNonSteadyPath( \( y^d_i \) )
- if \( \forall j \in P_d \) isSteady( \( y^d_j \) ) then
  - for \( j \in P_d \) do
    - DecreaseSteadyPath( \( y^d_j \) )
  - end for
- end if

MIRTO performs a rate decrease over all paths if they are all sensed in *steady state* this allows to leave local optima and adapt to new network conditions.

**D. Limited backlog at sources**

Assuming demands have infinite backlog, every ACK triggers a steady rate increase. If the demand is rate limited, it frequently happens that the source has no sufficient backlog to send. The effective rate increase is given by the minimum between the potential rate and the actual backlog. This regularly happens in the current Internet in case the application is not able to feed the TCP sending buffer at the network rate. Other important scenarios exhibit such behaviors, like multi-hop TCP and split TCP [4]. In all cited cases the rate increase will be sublinear for AIMD up to the peak rate of the source. In [4] authors study analytically this phenomenon in case the slow application is just an upstream slower TCP connection that might not feed the second one at the requested rate. This phenomenon cannot be analyzed within the optimization framework (1), because it neglects time evolutions and can only capture utility saturation, thus neglecting slow convergence to the peak rate. This does not mean that flow controllers can be effectively employed for infinitely backlogged sources only, but that rate limited transmitters would probe the network state at a slower rate. This also means that the fairness criterion of rate limited flows is slightly changed with respect to the original design.

**E. Implementation and testing**

We have implemented MIRTO in a service overlay with two kinds of nodes: access and relay. Nodes can be installed next to core and access routers that employs a transport network overlay that routes traffic from the ingress to the egress employing MIRTO. We present here a simple test that we run on the Grid5000 platform [2] emulating a network with the Abilene topology in fig2. Links are emulated using IP tunnels with shaped rate at 10Mbps. We present here a test with 30 demands flowing through the network with a hotspot traffic matrix, being the hot-spot node number 5, with 30 demands from nodes number 2,6,10. Two flows (flow 1 and flow 2) have access rate of 10Mbps while all the others between 250kpbs and 1Mbps. In this test we want to show that it is possible to implement multi-path routing in a real protocol serving realistic traffic demands with heterogeneous access rates. In fig.5 we report only flow 1 and 2 which share one bottleneck along their secondary paths (path 2). Flow 2 arrives 5 minutes after flow 1 and leave some minutes later. Low rate flows are routed along their shortest path and do not employ multi-path routing which is, on the contrary, employed by flow 1 and 2. Notice also that secondary path are probed with a certain delay that can be seen in the figure. MIRTO dynamics will be studied in the following section in more detail. This very simple test encourages us to proceed towards an experimental phase that is object of our incoming research.

**V. Fluid modeling of routing and flow control**

We model each source using a fluid representation of the sending rate \( y^d_i(t) \) along each path \( i \) for a given demand \( d \). Therefore, rates’ evolution is described through a system of deterministic ODEs along the line of classical fluid models of TCP [22]. For all \( y^d_i, i \in P_d, d \in D \)

\[
\frac{dy^d_i(t)}{dt} = \alpha y^d_i(t) - \beta y^d_i(t)\phi_i(t - R_i) - \gamma \sum_{j \in P_d} y^d_j(t)\zeta_i(t - R_i)
\]

(2)

Let us illustrate each term in detail.
1) Increase term: the first term, at the right member of (2) accounts for the additive increase of $y_d^i(t)$ over time with slope $\alpha/R_d^i(t)^2$ where $\alpha$ is the increase parameter (= 1 in TCP Reno) and $R_d^i(t)$ the round trip time experienced by flow $d$ along path $i$, i.e. transmission plus queuing plus propagation delay. The increase takes place when the path is selected, according to the decision function $\psi_i(t)$.

$$
\psi_i^d(t) = \begin{cases} 
\prod_{j \in P_u \setminus \tilde{P}_d} \mathbb{1}\{S_i^d < S_j^d\} & \text{if } i \in \tilde{P}_d \\
\prod_{j \in P_u \setminus \tilde{P}_d, j \notin i} \mathbb{1}\{S_i^d < S_j^d\} & \text{if } i \in P_u \setminus \tilde{P}_d.
\end{cases}
$$

(3)

where $\tilde{P}_d$ defines the set of all paths in “steady state” and $S_i^d(t)$ is a path cost measure defined as

$$
S_i^d(t) = \left\{ \begin{array}{ll}
\sum_{k \in L_i^d} \frac{1}{C_i} & \forall k \in L_i^d, Q_k(t) < B_k \\
\infty & \exists k \in L_i^d, Q_k(t) = B_k.
\end{array} \right.
$$

(4)

$Q_k(t)$ denotes the size of queue $k$ at time $t$. As one can remark from (3), the decision function differently acts on paths that are in transitory or in steady state (the definitions of these terms will be precisely defined later). The path selection described in 3, 4 obeys to these rules:

- As long as path $i$ is in steady state, $y_d^i$ grows if and only if $i$ is the minimum cost path among all.
- When path $i$ is transient, $y_d^i$ grows if $i$ is the best path among all transient paths.

Path $i$ is assumed to be congested if at least one link $k, k \in L_i^d$ is in saturation, i.e. $Q_k(t) = B_k$ being $B_k$ the buffer size and $L_i^d$ the link set of demand $d$ over path $i$. The queue models will be presented later. The cost associated to a non congested path $i$, $S_i^d$ is given by the sum of the inverse of capacities of links in $L_i^d$ (4), i.e. the sum of the link costs.

2) Decrease terms: the second and the third term at the right member of (2) account for the rate decrease. A congestion notification on a link within path $i$ is indicated by a rate reduction of $\beta y_d^i(t)$ ($\beta = 1/2$ in TCP Reno). $\phi_d^i(t)$ indicates the occurrence of congestion within path $i$ for flow $d$, as

$$
\phi_d^i(t) = 1 - \prod_{k \in L_i^d} \mathbb{1}\{Q_k(t) < B_k\}
$$

regardless of the queue model we will consider in the following section.

In addition to the multiplicative decrease of $y_d^i(t)$ our algorithm introduces a coordinated reduction of the rate, through the term $\gamma \sum_{j \in P_u} y_d^j(t)$, proportional to the total rate of flow $d$, when all paths are either congested or in steady state, as expressed by $\zeta^d(t)$,

$$
\zeta^d(t) = \prod_{j \in P_u \setminus \tilde{P}_d} \mathbb{1}\{S_j^d(t) = \infty\}
$$

$\zeta^d(t)$ is conventionally set to one if $P_u \setminus \tilde{P}_d \neq \emptyset$.

The decrease parameter $\gamma$ quantifies the level of coordination between all paths of a given flow $d$ as it intervenes on all paths jointly. Let us get back to the definition of “steady state”. It is worth observing that the increase/decrease dynamics of the rate lead to an oscillatory stationary regime already observed in TCP Reno under drop tail. Thus, we define a flow to be stationary as long as the variations of its mean value remain bounded by a constant $\varepsilon$, i.e. $|\tilde{y}_d^i(t) - \hat{y}_d^i(t)| < \varepsilon$. $\tilde{y}_d^i(t)$ denotes the exponential moving average up to time $t$ with smoothing parameter $T_d^i$, taken proportional to $R_i(t)$, $\tilde{y}_d^i(t)/dt = -[\tilde{y}_d^i(t) - \tilde{y}_d^i(t)]/T_d^i$.

A. Queue models

The queue models under study are FIFO with drop tail and FQ with drop from the longest queue first (DLQF). FQ is used as a neutral scheduler that does not realise any particular form of fairness because it delegates this issue to end-to-end protocols. FQ, on the contrary imposes at every link max-min fairness among all flows crossing it ([11], [12]). These two schedulers allows us to model the two cases of coordinated and uncoordinated multipath routing.

1) FIFO: The time evolution of $Q_k(t)$ follows:

$$
\frac{dQ_k(t)}{dt} = A_k(t) - D_k(t) - L_k(t)
$$

(5)

where each term is defined according to

$$
A_k(t) = \sum_{d \in D} r_d^k y_d^i(t)
$$

$$
D_k(t) = C_k \mathbb{1}\{Q_k(t) > 0\}
$$

$$
L_k(t) = (A_k(t) - C_k)^+ \mathbb{1}\{Q_k(t) = B_k\}
$$

The arrival rate $A(t)$ is the superposition of all flow rates routed through the queue $Q(t)$. The departure rate $D(t)$ is given by the link capacities when the queue is non empty and zero otherwise. The loss rate $L(t)$ is given by the exceed rate $A(t) - C$ in congestion: $A(t) > C, Q(t) > B$, with $B$ the storage capacity. $r_d^k$ is one if flow $d$ is routed through link $k$, zero otherwise.

2) FQ: Time evolution of $Q_k^d$, the per flow occupation, is driven by the following set of equations

$$
\frac{dQ_k^d(t)}{dt} = A_k^d(t) - D_k^d(t) - L_k^d(t), \quad \forall d \in D
$$

(6)

where each term is defined according to

$$
A_k^d(t) = r_d^k y_d^i(t)
$$

$$
D_k^d(t) = C_k \sum_{d' > d} r_d^{k'} y_d^{k'}(t) \mathbb{1}\{Q_k^d(t) > 0\}
$$

$$
L_k^d(t) = (A_k^d(t) - C_k)^+ \mathbb{1}\{d = \arg \max_d Q_k^d(t), \sum_{d'} Q_k^{d'}(t) = B_k\}
$$

$A^d(t)$ is the flow rate of flow $d$, $D_k^d(t)$ the rate scheduled to flow $d$ and $L_k^d(t)$ the loss rate experienced by flow $d$, according to the longest queue drop policy.

If the demand is bottlenecked at the source, equation (2) is slightly modified with an additional term at the right member: a decrease term equal to $\alpha/R_d^i(t)^2$ over the most expensive path with respect to the previously described definition of cost. This means that, as a path is congested, the same amount of rate is moved from a bad path to a better one. The presence of peak rates does not allow a flow $d$ to probe a path indefinitely as this would not result in any benefit.
TABLE I
SUMMARY TABLE WHERE FLOWS HAVE NO RATE LIMITATIONS: FIFO AND FQ (IN BRACKETS). LINK1=(3,5), LINK2=(6,5), LINK3=(8,5).

<table>
<thead>
<tr>
<th>Flow</th>
<th>Throughput</th>
<th>Link1</th>
<th>Link2</th>
<th>Link3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIRTO</td>
<td>100 (50)</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TEXCP</td>
<td>109 (108)</td>
<td>-</td>
<td>24 (25)</td>
<td>-</td>
</tr>
<tr>
<td>TEXCPSP</td>
<td>100 (108)</td>
<td>-</td>
<td>24 (25)</td>
<td>-</td>
</tr>
<tr>
<td>TRUMP</td>
<td>100 (98)</td>
<td>-</td>
<td>16 (25)</td>
<td>-</td>
</tr>
</tbody>
</table>

TABLE II
SUMMARY: NETWORK USAGE AND FLOW THRUPUT, FIFO AND FQ (IN BRACKETS).

<table>
<thead>
<tr>
<th>Flow</th>
<th>Throughput</th>
<th>Link1</th>
<th>Link2</th>
<th>Link3</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIRTO</td>
<td>600 (939)</td>
<td>300 (299)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TEXCP</td>
<td>720 (854)</td>
<td>300 (300)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TEXCPSP</td>
<td>731 (939)</td>
<td>300 (299)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TRUMP</td>
<td>950 (866)</td>
<td>300 (278)</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

B. Numerical evaluations

In this section we study the functioning of MIRTO as previously described by means of numerical evaluations of the equations derived in section V. We compare it to other two recently proposed algorithms: TEXCP and TRUMP that can be modeled by the fluid equations that we report in appendix B. The solution is obtained by solving the system of ODEs representing source rates (2) and queues (5, 6). We use the method of Runge-Kutta of the 4-th order to solve the problem.

Figure 2 depicts the network topology used for the analysis. All link capacities are set to C=100Mbps, queue limit is 10 packets and propagation delays negligible. RTT is then given by transmission plus queuing delay. We consider a hot-spot scenario where differences between FIFO and FQ scheduling can be detected. Three sources, nodes number 2, 6 ,10, send traffic to the same destination, node 5 (flow 1,2,3 respectively). Link (3,5), (6,5), (8,5) are bottlenecks.

We chose as path cost the sum of the inverse of the link capacities and, as they are the same, path ranking is made by hop counting. We analyze two different traffic patterns: a first one where all flows have infinite backlog and a second where one out three is access limited to 50Mbps. Results include calculations for scenarios with the two queue policies: FIFO with drop tail and FQ with DLQF. All parameters used for the presented numerical analysis are summarized in table IV. They all have been set following the advices included in the original papers to achieve stability and fast convergence. Let us consider fig.6 that depicts the time evolution of the rate of flow 1 through the available paths, in presence of FIFO scheduling in the nodes. Table I is also helpful to see the global picture.

Observe that in table I TEXCP appears in two versions whether or not it implements a feature that gives priority to shortest paths (TEXCPSP see [15] and appendix B. All flows get an aggregate thruput of 100Mbps. MIRTO employs only shortest paths and probes secondary routes continuously without being able to get more share. TEXCP and TRUMP behave similarly: all flows split their rate among three paths even if they could...
be satisfied using a single route. From table II, we see that all protocols get the same thruput while the total network utilization is higher for TRUMP and TEXCP. MIRTO consumes less resources and TEXCPSP also get improvements.

TRUMP and TEXCP converge fast tough all parameters need to be tuned ad hoc for each particular scenario. Notice also that secondary path in MIRTO increase its rate sublinearly as they do not increase at every ACK reception. Our experience on multiple scenarios (non reported for lack of space), says that there is no simple rule to tune TRUMP and TEXCP but they are very much dependent on the network setup. The use of FQ exhibits a waste of resources for the same thruput. As we have remarked (remarks 2,3) in section III, in large scenarios path uncoordination, in spite of being suboptimal, does not have a measurable negative impact. Nevertheless, suboptimality shows up in smaller examples. Table III summarizes additional results for the scenario where flow 2 is rate limited (see table II for global statistics). In this scenario we see that resources freed by flow 2 are exploited by other flows that fill longer paths with consequent larger costs for the network. Notice also that FIFO and FQ have more similar performance now. In all cases MIRTO consumes less resources for the same thruput.

### VI. Discussion and Conclusions

In many applications scalability concerns arise if multipath routing is implemented at flow level. Considering that per-flow state would be maintained at edge only, this might not actually be an issue thanks to little multiplexing at access nodes.

Another possibility is to aggregate all flows as origin-destination pairs, regardless of their characteristics. This might be harmful for low rate flows as they would be routed together with higher rate flows. As we have shown in section III for large problems and in V-B for little scenarios, the use of secondary paths is not recommended for low rate ones. Therefore, aggregation makes sense only for homogeneous setup. This is the case for service overlays and virtual networks transporting a common service.

The use of FQ is still recommended in practice even though suboptimal in theory. This is our view on the issue especially because FQ assures predictable performance in presence of very heterogeneous cases where every flow might employ a different protocol. The main reason to use FQ is that it does not rely on source conformant to the same fairness criterion, which is often the case. Nevertheless FQ needs a proper definition of flow for all nodes which might be non trivial.

Dynamic yet stable robust routing sensibly ameliorates performance in a large number of networking applications: intradomain TE, routing in multi-AS ISPs, service overlays, virtual networks. In the last few years there has been significant research effort to overcome the impasse on network routing that dates back to unsuccessful early experiences in the very beginning of the Internet. In this paper we have placed the subject in a realistic traffic context in order to solve some issues left open in recent protocol propositions. In this paper we have proposed a protocol MIRTO, from the conception from optimization to the implementation in a protocol passing through the analysis of its dynamic using fluid ODEs that allowed the comparison with other two recent propositions.

### References

[9] He, J.; Bresler, M.; Chiang, M. and Rexford, J. Towards Robust Multi-Path TCP: A Large-Scale Field Trial. In proc. of ACM SIGCOMM'05
APPENDIX A

LINEAR PROGRAM FORMULATION

In this appendix we consider the problem (1) and select one particular fairness criterion: max-min. We write this problem as an iterative linear program assuming linear costs $V$ (see [24] for more details). At each iteration a linear sub-problem is solved giving, at optimum, the same network flow share $z$ to the most constrained demands. This share is the maximum bandwidth that can be allocated to the most constrained demand. The network graph $G$ is reduced to $\tilde{G}$ through the following transformation: $C_{ij} = C_{ij} - \sum_{d \in D} x^d_{ij},$ i.e. capacities are replaced by residual capacities after the allocation of this share of bandwidth.

Also $u_i = \sum_{j \in N} \sum_{d \in D} x^d_{ij}.$

If $\tilde{C}_{ij} = 0$ the link is removed from the graph. Like capacities, every demand $p_i^d$ is reduced to $\tilde{p}_i^d = p_i^d - z,$ i.e. demands’ peak rates are replaced by their residuals after the bandwidth share allocation.

If $\tilde{p}_i^d = 0$ the demand is removed from the demand set because it has been completely satisfied. These iterations might be interpreted as the classical filling procedure to obtain max-min fair sharing, which essentially is a gradient algorithm. The sub-problem is formalised as follows: $\text{SUBLP}(G, \tilde{G}, D):$

\[
\begin{align*}
\text{maximize} & \quad z - \sum_{i,j \in N} V(\sum_{d \in D} x^d_{ij}/C_{ij}) \\
\text{subject to} & \quad \sum_{k \in N} x^d_{ki} - \sum_{j \in N} x^d_{ij} = \phi_i(z) \quad \forall i \in N, d \in D \\
& \quad \sum_{d \in D} x^d_{ij} \leq C_{ij} \quad \forall i, j \in L, d \in D \\
& \quad z \leq p^d \quad \forall d \in D.
\end{align*}
\]

In case of path uncoordination two constraints are changed, and $f^d$ is introduced, i.e. the maxflow of demand $d$ with respect to the capacitated graph $G.$

\[
\begin{align*}
\sum_{k \in N} x^d_{ki} - \sum_{j \in N} x^d_{ij} = \phi_i(f^d z) \quad \forall i \in N, d \in D \\
\sum_{d \in D} x^d_{ij} \leq C_{ij} \quad \forall i, j \in L, d \in D \\
z f^d \leq p^d \quad \forall d \in D.
\end{align*}
\]

In this appendix we write the fluid equations that describe the dynamics of TRUMP and TEXCP. We keep, for each algorithm, the same notation of the corresponding original paper in order to help the reader comparing these equations with the protocol definition. The following notation is valid only within the scope of this appendix and must not be compared with that of the rest of the paper.

Definition 5: TRUMP fluid equations.

The dynamic of TRUMP is given by the following ODEs at link $l,$

\[
\begin{align*}
p_i(t) &= p_i(0) + \max \left(0, -f^d_i - \beta(C_l - \sum_d q^d_i(t))dt\right), \\
d_i(t) &= \frac{\sum_{j < l} q^d_j(t)}{C_{ij}} - \frac{\sum_{k > l} q^d_k(t)}{C_{ki}}, \quad s_i(t) = p_i(t) + d_i(t)
\end{align*}
\]

and at source $d$ and being $R_i$ the round trip time of path $i$

\[
\frac{dg^d_i}{dt} = -\gamma \sum_{i} y^d_i(t) - \gamma \sum_{i} s_i(t - R_i)
\]

Definition 6: TEXCP fluid equations.

Consider at link $l,$ $N_l$ the number of flows, $Q_l(t)$ queue length at link. Also $u_i = \sum_{j \in N} \sum_{d \in D} x^d_{ij}$

\[
\frac{dx^d_i}{dt} = \frac{z^d_i}{\sum_{j \in N} x^d_{ij}(t)} \left(\sum_{j < l} \sum_{d} y^d_{ij}(t)u_{ij}(t - R) - u_i(t - R)\right) + \epsilon \quad u_i = u_{\text{min}}
\]

Define also $\frac{dg^d_i}{dt} = \delta_+ - \delta_i g^d_i(t - R)$ and $\phi_i = \alpha(C_l - \sum_d q^d_i(t)) - \beta Q_l(t) (\delta^+, \delta^-) = \left(\phi_i, \phi_i\right)$ if $\phi_i \geq 0$ and $(\delta^+, \delta^-) = \left(0, \frac{\phi_i}{\sum_d q^d_i}\right)$ if $\phi_i < 0.$ In case TEXCP gives priority to the shortest path (TEXCPSP) an additional variable is added:

\[
v^l_i = 2R_i^2 \gamma^+ \quad \gamma^+ = \frac{\sum_{j} x^d_{ij}}{\sum_{i} x^d_{ij}}
\]