Path Selection and Multipath Congestion Control

Peter Key
Microsoft Research Cambridge
7 J J Thomson Avenue
Cambridge CB3 0FB, UK
peter.key@microsoft.com

Laurent Massoulié
Thomson Technology Paris Laboratory
1, rue Jeanne d'Arc
92443 Issy-les-Moulineaux cedex - France
laurent.massoulie@technicolor.com

Don Towsley
Department of Computer Science
University of Massachusetts
Amherst, MA 01003, USA
towsley@cs.umass.edu

ABSTRACT

In this paper we investigate the benefits that accrue from the use of multiple paths by a session coupled with rate control over those paths. In particular, we study data transfers under two classes of multipath control, *coordinated control* where the rates over the paths are determined as a function of all paths, and *uncoordinated control* where the rates are determined independently over each path. We show that coordinated control exhibits desirable load balancing properties; for a homogeneous static random paths scenario, we show that the worst-case throughput performance of uncoordinated control behaves as if each user has but a single path (scaling like $\log(\log(N))/\log(N)$ where N is the system size, measured in number of resources). Whereas coordinated control yields a worst-case throughput allocation bounded away from zero. We then allow users to change their set of paths and introduce the notion of a Nash equilibrium. We show that both coordinated and uncoordinated control lead to Nash equilibria corresponding to desirable welfare maximizing states, provided in the latter case, the rate controllers over each path do not exhibit any RTT bias (as in TCP Reno). Finally, we show in the case of coordinated control that more paths are better, leading to greater welfare states and throughput capacity, and that simple path reselection polices that shift to paths with higher net benefit can achieve these states.

1. INTRODUCTION

Multipath routing has received attention recently [2, 6, 14, 5, 21]. Furthermore, combining multipath routing with rate control is implicitly used by several Peer-to-peer (P2P) applications. Most relevant to us is Bittorrent [4], which maintains a number of, typically four, active connections to other peers with an additional path periodically chosen at random together with a mechanism that retains the best paths (as measured by throughput).

The basic setting of multipath coupled with rate control is as follows. A source and destination pair in a network is given a set of possibly overlapping paths connecting them. The pair then chooses a subset to use and the rates at which to transfer data over those paths. This scenario is illustrated in Figure 1(a). Note that the P2P example described above falls into this formulation once one includes a fictitious source feeding data through peers to the intended receiver, as in Figure 1(b). Some natural questions arise:

- How many paths are required? And does it suffice, as with the above P2P application, to use a subset of the paths? Opening multitudinous TCP connections has negative systems performance implications, hence there are incentives to keep the number of connections small.
- P2P applications use independent *uncoordinated* TCP rate control mechanisms over each active path as this is straightforward to implement and requires no change to the network. However, starting from first principles, mechanism design produces a *coordinated* control mechanism where the rates over each path are determined as a function of all of the paths. How does an uncoordinated control mechanism perform relative to a coordinated control mechanism? This is important because the latter requires a revised transport layer protocol or a careful application layer solution whereas the former is easily implemented using TCP.

The motivating application scenario is of data transfers over a network, where the transfers are long enough to allow performance benefits for multipath routing, although our results apply more generally to situations where there are alternative resources which can help service a demand, and where the demand is serviced using some form of rate control. We assume that demand is fixed, and each user[†] attempts to optimize its performance by choosing appropriate paths (resources), where

[†]We use the term 'user' as a convenient shorthand for a user, or the software or algorithm a user or end-system employs.

The original version of this paper is entitled "Path Selection and Multipath Congestion Control" and was published in Proceedings of IEEE Infocom 2007, May 2007 by IEEE.

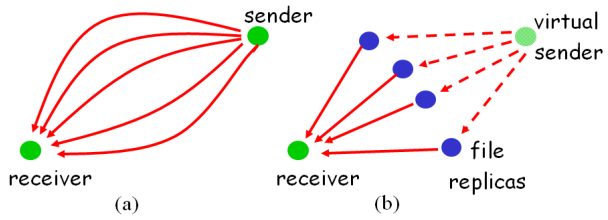

Figure 1: (a) A canonical multipath example. (b) A BiTorrent example where a receiving peer receives data from four peers. A virtual sender has been included to show the relationship to canonical multipath.

the rate control algorithm is fixed. More precisely, we assume that the rate control is implicitly characterized by a utility maximization problem [20], where a particular rate control algorithm (e.g. TCP Reno) maps to a particular (user) utility function [9], and users selfishly seek to choose paths in such a way as to maximize their net utility. Within this optimization framework, a coordinated controller is modeled by a single utility function per user, whose argument is the aggregate rate summed over paths, whereas an uncoordinated controller has a utility function per path and the aggregation is over all of the utility functions.

Key to the usefulness of multipath rate control is its ability in the hands of users operating independently of each other to balance the load throughout the network. We illustrate this for a particular scenario, where the paths chosen are fixed and static, but chosen at random from a set of size N. We focus on the worst-case allocation, which is a measure of the fairness of the scheme. In the uncoordinated case, the worst case allocation scales as $\log(\log(N))/\log(N)$ independent of the number b of paths chosen. In contrast, in the coordinated case where each user can balance its load across the$b$ paths available to it, provided there are two or more, the worst-case allocation is bounded away from zero. This demonstrates that

1. coordinated control balances loads significantly better than uncoordinated when paths are fixed;
2. coordinated improves on greedy least-loaded resource selection, as in Mitzenmacher [16], where the least-loaded selection of b resources scales as $1/\log(\log(N))$ for $b > 1$.

Effectively, coordinated control is able to shift the load amongst the resources, and with each user independently balancing loads over no more than two paths, able to utilize the resources *as if* global load balancing was being performed.

This raises the question of how users should be assigned a set of paths to use. One natural path selection mechanism is to allow users to make their own choices. We study this as a game between users and consider a natural notion of a Nash equilibrium in this context, where users seek to selfishly maximize their own net utilities. We find that when users use coordinated controllers, the Nash equilibria coincide with welfare-maximizing social optima. When we consider uncoordinated controllers, then the results depend on whether or not the controllers exhibit RTT bias (like TCP) or not. When they do not exhibit RTT bias, the Nash equilibria also coincide with welfare-maximizing social optima. Otherwise they need not.

We show that increasing the number of paths available to a source destination pair is desirable from a performance perspective. However, the simultaneous use of a large number of paths may not be possible. We show that this does not pose a problem as simple path selection policies which combine random path resampling with moving to paths with higher net benefit lead to welfare maximizing equilibria and also increase the throughput capacity of the network. In fact such a policy does as well as if each user uses all of the available paths simultaneously.

In summary, we shall provide some partial answers to our initial questions.

- In a large system, provided users re-select randomly from the sets of paths and shift to paths with higher net benefit, they rely on a small number of paths and do as well as if they were fully using all available paths.
- Coordinated control has better fairness properties than uncoordinated in the static case. When combined with path reselection, uncoordinated control only does as well as a coordinated control if there is no RTT bias in the controllers.

We conclude the paper with some thoughts on how multipath rate control might be deployed.

2. THE MULTIPATH FRAMEWORK

The standard model of the network is as a capacitated graph $G = (V, E, C)$ where V represents a set of end-hosts or routers, E is a set of communication links and each link has a capacity, say in bits per second, C_l, $l \in E$. In addition a large population of sessions perform data transfers over the network. These sessions are partitioned into a set of session classes $\mathcal{S}$ with N_s sessions in class $s \in \mathcal{S}$. Associated with class s is a source σ_s, a destination d_s, and a set of one or more, possibly overlapping paths, $\mathcal{R}(s)$ between the source and destination that is made available to all class s sessions. Finally, we associate an increasing, concave function with each session class, $U_s(x)$, which is the utility that a class s session receives when it sends data at rate $x > 0$ from source to destination. Now, exactly how this utility is used and the meaning of x depends on whether we are concerned with coordinated or uncoordinated control. We will shortly describe each of these in turn.

A discussion of how utility functions can be used to model standard TCP Reno is given in [15]. The so-called weighted alpha-fair utility functions given by

$$U_r(x) = \begin{cases} w_r \frac{x^{1-\alpha}}{1-\alpha} & \text{if } \alpha \neq 1 \\ w_r \log(x) & \text{if } \alpha = 1 \end{cases} \quad (1)$$

were introduced in [17], and are linked to different notions of fairness. For example, $\alpha = 1$ corresponds to

(weighted) proportional fairness [8], and $\lim \alpha \to \infty$ to max-min fairness. TCP's behavior is well approximated by taking $\alpha = 2$ and $w_r = 1/T_r^2$, where T_r is the round trip time for path r, in the following sense: TCP achieves the maximum aggregate utility, for given paths and link capacities, for the corresponding utility functions U_r.

The set of paths available to a class s session can potentially be very large. Hence a session will likely use only a small subset of these paths. We assume for now that every class s session uses exactly b_s paths. Let c denote a subset of $\mathcal{R}(s)$ that contains b_s paths and $\mathcal{C}(s)$ the set of all such subsets of paths, $\mathcal{C}(s) = \{c : c \subset \mathcal{R}(s) \wedge |c| = b_s\}$. Let N_c denote the number of class s sessions that use the set of paths $c \in \mathcal{C}(s)$, $s \in \mathcal{S}$, and hence $N_s = \sum_{c \in C(s)} N_c$. Last, let N_r denote the number of sessions that use path $r \in \mathcal{R}(s)$, $N_r = \sum_{c \in \mathcal{C}(s)} \mathbf{1}(r \in c))N_c$.

Associated with each class s session is a congestion controller (rate controller) that determines the rates at which to send data over each of the b_s paths available to it. We now distinguish between coordinated and uncoordinated control.

Coordinated control. Given a set of paths c, a coordinated controller actively balances loads over all paths in c, taking into account the states of the paths. Our understanding of and ability to design such controllers relies on a significant advance made by Kelly [8], which maps this problem into one of utility optimization. In the case of coordinated congestion control, the objective is to maximize the 'social welfare', that is to

$$\text{Maximize} \sum_{s \in \mathcal{S}} \sum_{c \in \mathcal{C}(s)} N_c U_s\big(\sum_{r \in c} \lambda_{cr}\big) \tag{2}$$

over $(\lambda_{cr} \geq 0)$ subject to the capacity constraints

$$\sum_{r \ni l} \sum_{c \ni r} N_c \lambda_{cr} \leq C_l, \qquad l \in E \tag{3}$$

where λ_{cr} is the sending rate of a class s session that is using path r in $c \in \mathcal{C}(s)$. We will find it useful to represent the total rate contributed by class s sessions that use path $r \in \mathcal{R}(s)$ as $\Lambda_r = N_c \sum_{c \ni r} \lambda_{cr}$, and the aggregate rate achieved by a single s session over all paths in c as $\lambda_c = \sum_{r \in c} \lambda_{cr}$.

Note that in the absence of restrictions on the number of paths used, $\mathcal{C}(s) = \mathcal{R}(s)$, and the optimization can be written

$$\text{Maximize} \sum_{s \in \mathcal{S}} N_s U_s\big(\sum_{r \in \mathcal{R}(S)} \lambda_r\big) \tag{4}$$

subject to the capacity constraints. We shall see later in Section 5 that by using random path reselection the solution to (2) actually solves (4), and hence give conditions for when the restriction to using a subset of paths of limited size imposes no performance penalties.

More generally, we can replace the hard capacity constraints[‡] by a convex non-decreasing penalty function Γ. In the context of TCP, this penalty function can be thought of as capturing the signaling conveyed by packet losses or packet marking (ECN, [19]) by the network to the sessions when link capacities are violated. Under this extension, the coordinated control problem transforms to

$$\text{Maximize} \sum_{s \in \mathcal{S}} \sum_{c \in \mathcal{C}(s)} N_c U_s\big(\sum_{r \in c} \lambda_{cr}\big) - \Gamma(\{\Lambda_r\}). \tag{5}$$

There are many ways to approach the problem of designing controllers that solve these problems, but a very natural one is suggested by the TCP congestion control, which solves this variation of the above problem when each session is restricted to a single path (see [11]).

Uncoordinated control. As mentioned earlier, uncoordinated control corresponds to a session with path set c executing independent rate controllers over each path in c. This is easily done in the current Internet by establishing separate TCP connections over each path. The ease in which uncoordinated control can be implemented motivates our study of it. In Kelly's optimization formulation this corresponds to solving the following problem,

$$\text{Maximize} \sum_{s \in \mathcal{S}} \sum_{c \in \mathcal{C}(s)} N_c \sum_{r \in c} U_r(\lambda_{cr}) \tag{6}$$

over non-negative λ_{cr} subject to the capacity constraints (3). As above, by analogy with (5) the constraints can be generalized to reflect the signaling used by a controller such as TCP. Note the difference between this formulation and that for coordinated control. In the case of the latter, the utility is applied to the *aggregate sending rate* whereas in the case of the former, the utility is evaluated on each path and then summed over all paths. Note also that really we have written U_r instead of U_s for the uncoordinated controller, to reflect the fact that the congestion control may differ across different paths (as is the case with TCP whose allocation depends on the RTT of the path).

The functions above are strictly concave and are being optimized over a convex feasible region. Hence the problems admit to unique solutions in terms of aggregate per class rates, even though distinct solutions may exit

3. LOAD BALANCING PROPERTIES OF MULTIPATH

Multipath has been put forward as a mechanism that, when used by all sessions can balance traffic loads in the Internet. It is impossible to determine whether this is universally true. However, we present in this section a simple scenario where this issue can be definitively resolved. We consider a simple scenario where there are N resources with unit capacity ($C_l \equiv 1$).

To provide a concrete interpretation, the resources can be interpreted as servers, or as relay or access nodes — see Figure 2. There are aN users. Each user selects b resources at random from the N available, where b is an integer larger than one (the same resource may be sampled several times). We shall look at the worst case

[‡]The hard constraints in (3) can be written as the sum of penalty functions, each of which is a step function $\Gamma_l(x)$, with $\Gamma_l(x) = 0$ if $x \leq C_l$ and ∞ otherwise

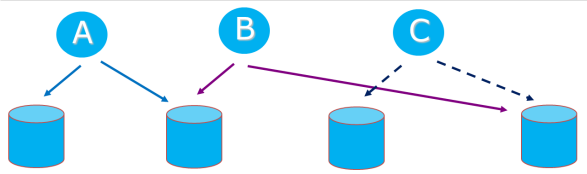

Figure 2: Load balancing example: there are N servers, aN users and each selects $b > 1$ servers at random.

rate allocation of users in two scenarios. In the first scenario, users implement uncoordinated multipath congestion control where there is no coordination between the b distinct connections of each user. Thus, a connection sharing a resource handling X connections overall achieves a rate allocation of exactly $1/X$. In the second scenario, each user implements coordinated multipath congestion control.

We take the worst-case user rate allocation (or throughput), as the load balance metric. One can show [13] that the more 'unfair' the allocation, the greater the expected time to download a unit of data.

3.1 Uncoordinated congestion control

Denote by λ_i the total rate that user i obtains from all its connections. In the case of uncoordinated congestion control, we can show that the worst case rate allocation, $\min \lambda_i$ decreases like $b^2 \log(\log N)/\log N$ as N increases. This is to be compared with the worst case rate allocation that one gets when $b = 1$, that is when a single path is used: from classical balls and bins models [16], this also decreases as $\log(\log(N))/\log(N)$ as N increases. It should come as no surprise that using more than two paths exhibits the same asymptotic performance as using only one path; there is no potential for balancing load within the network when all connections operate independent of each other. A formal statement and proof of this result can be found in [11].

3.2 Coordinated congestion control

Here we assume as before that there are aN users, each selecting b resources at random, from a collection of N available resources. Denote by λ_{ij} the rate that user i obtains from resource j, and let $\mathcal{R}(i)$ denote the set of resources that user i accesses. In contrast with the previous situation, we now assume that the rates λ_{ij} are chosen to maximize:

$$\sum_{i=1}^{aN} U\Big(\sum_{j\in\mathcal{R}(i)} \lambda_{ij}\Big) \text{ subject to } \sum_i \lambda_{ij} \le 1 \quad \text{for all } j, =$$

for some concave utility function U.

An interesting property of this problem is that the set of $\{\lambda^*_{ij}\}$ that solves the above optimization is insensitive to the choice of utility function U so long as it is concave and increasing. Moreover, this insensitivity implies that the optimal aggregate user rates (λ^*_i) correspond to the max-min fair rate allocations [3, Sec. 6.5.2]. Simply stated a rate allocation (λ^*_i) is said to be *max-min fair* if and only if an increase of any rate $\lambda^*_{i_0}$ must result in the decrease of some already smaller rate. Formally, for any other feasible allocation (x_i), if $x_i > \lambda_i$ then there must exist some j such that $\lambda^*_j < \lambda^*_i$ and $x_j < \lambda_j$. The above statements are easily verified by checking that the max-min fair allocation satisfies the Karush-Kuhn-Tucker conditions associated with the above optimization problem.

This leads to the following result:

THEOREM 3.1. *Assume there are N resources, and aN users each connecting to b resources selected at random. Denote by $\{\lambda^*_i\}$ the optimal allocations that result. Then there exists $x > 0$, that depends only on a and b, such that:*

$$\lim_{N\to\infty} \mathbf{P}\left(\min_i \lambda^*_i \ge x\right) = 1. \tag{7}$$

The style of the proof has wide applicability and we outline it here: first, an application of Hall's celebrated marriage theorem shows that the minimum allocation will be at least x provided that any set of users (of size n say) connect to at least x times as many servers (nx servers). If this condition is satisfied, the allocation (λ^*_i) will exceed x; hence it is sufficient to ensure that Hall's condition is met with high probability for all non-empty subsets of $1, \ldots, aN$. Using the binomial theorem and the union bound yields an upper bound on the probability the condition fails to hold, and then Stirling's approximation is used to approximate this bound.

This result says that the worst case rate allocation is bounded away from zero as N tends to infinity, i.e., it is $O(1)$ in the number of resources N. Thus coordinated control exhibits significantly better load balancing properties than does uncoordinated control. It is also interesting to compare this result to the result quoted by Mitzenmacher et al. [16], which says that if users arrive in some random order, and choose among their b candidate resources one with the lowest load, then the worst case rate scales like $1/\log(\log(N))$, which unlike the allocation under coordinated control, goes to zero as N increases. The difference between the two schemes is that in Mitzenmacher's scheme a choice has to be made immediately at arrival, which cannot be changed afterwards, whereas a coordinated controller actively and adaptively balances load over the b paths reacting to changes that may occur to the loads on the resources.

4. A PATH SELECTION GAME

In this section we address the following question. Suppose that each session is restricted to using exactly b paths each, taken from a much larger set of possible paths: what is the effect of allowing each user to choose its b paths so as to maximize the benefit that it receives? To answer this question, we study a *path selection game*. Here each session is a player that greedily searches for throughput-optimal paths. We characterize the equilibrium allocations that ensue. We show that the same equilibria arise with coordinated congestion control and uncoordinated congestion control provided that the latter does not introduce RTT biases on the different paths. Moreover, these equilibria correspond

to the optimal set of rates that solve problems (2) and (6), i.e., achieve welfare maximization. We shall use the models and notation of Section 2.

We shall restrict attention to when N_s is large, so that a change of paths by an individual player (session) does not significantly change the network performance. In game theory terms we are only considering *non-atomic* games.

4.1 Coordinated congestion control

For coordinated control, we use the model of Section 2, where the number of sessions N_s is fixed for all s, and introduce the following notion of a Nash equilibrium:

Definition 4.1: The non-negative variables N_c, $c \in \mathcal{C}(s)$, $s \in \mathcal{S}$, are a Nash equilibrium for the coordinated congestion control allocation if they satisfy the constraints $\sum_c N_c = N_s$, and moreover, for all $s \in \mathcal{S}$, all $c \in \mathcal{C}(s)$, if $N_c > 0$, then the corresponding coordinated rate allocations satisfy

$$\sum_{r \in c} \lambda_{cr} = \max_{c' \in \mathcal{C}(s)} \sum_{r \in c'} \lambda_{c'r}. \tag{8}$$

In other words, for each session (player), weight is only given to sets c that maximize the throughput for s. $\diamond$

We then have the following:

THEOREM 4.1. *At a Nash equilibrium as in Definition 4.1, the path allocations λ_r solve the welfare maximization problem* (4).

The proof follows since at a Nash equilibrium, type s players only use minimum 'cost' paths, which can be shown to coincide with the Kuhn-Tucker conditions of (4). This result says that a selfish choice of path-sets by end-users results in a solution that is socially optimal.

4.2 Uncoordinated control

We introduce the following notion of Nash equilibrium:

Definition 4.2: The collection of per path connection numbers N_r is a Nash equilibrium for selfish throughput maximization if it satisfies $\sum_r N_r = N_s$, and furthermore, the allocations (6) are such that for all $s \in \mathcal{S}$, all $r \in \mathcal{R}(s)$, if $N_r > 0$, then

$$\lambda_r = \max_{r' \in \mathcal{R}(s)} \Lambda_{r'}. \tag{9}$$

$\diamond$

The intuition for this definition is as follows: any class s session maintains a connection along path r only if it cannot find an alternative path r' along which the default congestion control mechanism would allocate a larger rate.

We then have the following result, whose proof is similar to that of Theorem 4.1.

THEOREM 4.2. *Assume that for each $s \in \mathcal{S}$, there is a class utility function U_s such that $U_r(x) \equiv U_s(x/b)$ for all $r \in \mathcal{R}(s)$. Then for a Nash equilibrium (N_r), the corresponding rate allocations (λ_r) solve the general optimization problem* (4).

To summarize: if i) the utility functions associated with the congestion control mechanism are path-independent, ii) users agree to concurrently use a fixed number b of paths, and iii) they manage to find throughput-optimal paths, that is they achieve a Nash equilibrium, then at the macroscopic level, the per-class allocations solve the coordinated optimization problem (4).

It is well known that the bandwidth shares achieved by TCP Reno are affected by the path round trip times. Thus the underlying utility functions are necessarily path dependent and the above result does not apply as i) fails to hold. As a consequence 'bad' Nash equilibria can exist. Indeed, a specific example is given in [11] where the preference of TCP for 'short-thin links' over 'fat-long-links' gives rise to a Nash equilibrium where the throughput is half of what could be achieved. If (ii) is relaxed, different uses have different 'market power', where those with larger b_s gain a large share, thus also creating 'bad' Nash equilibria in general.

5. MULTIPATH ROUTING WITH RANDOM PATH RESELECTION

In the previous section we explored the effect of allowing users to greedily select their set of paths (b in number) out of the set of all possible paths that are available to them. In this section we focus on two questions. The first regards how many paths, b_s, to allow each class s user so as to enhance its performance and that of the system. We establish a monotonicity result for coordinated control in order to address this question. The second question regards how to manage the overhead that may ensue due to the need for a user to balance load actively over a large number of paths. Possibly surprisingly, we will show that it suffices for a user to maintain a small set of paths, say two ($b = 2$), provided that it repeatedly selects new paths at random and replaces the old paths with these paths when the latter provide higher throughput. It is interesting to point out that BitTorrent uses a strategy much like this where it 'unchokes' a peer (tries out a new peer) and replaces the lowest performing of its existing four connections with this new connection if the latter exhibits higher throughput.

We will examine the above questions for both coordinated control and uncoordinated control. We begin with coordinated control.

5.1 Coordinated control

We begin by addressing the first question, namely how many paths are needed. Consider a network G that supports a set of flow classes $\mathcal{S}$ with populations $\{N_s\}$, and utility functions $\{U_s\}$. Let $\{\mathcal{R}(s)\}$ and $\{\mathcal{R}'(s)\}$ be two collections of paths for $\mathcal{S}$ that satisfy $\mathcal{R}(s) \subseteq \mathcal{R}'(s)$ for $s \in \mathcal{S}$ and suppose that each session applies coordinated control over these paths. Then for the problem (4)

$$\sum_{s \in \mathcal{S}} N_s U_s\Big(\sum_{r \in \mathcal{R}(s)} \lambda_{sr}\Big) \leq \sum_{s \in \mathcal{S}} N_s U_s\Big(\sum_{r \in \mathcal{R}'(s)} \lambda_{sr}\Big)$$

and hence performance increases when the path-sets increase, with performance measured by the optimal wel-

fare under the capacity constraints. This follows from the fact that a solution honoring the constraints on path sets $\{\mathcal{R}\}$ remains a feasible solution when the set of paths increases.

REMARK 5.1. *Although we have not shown strict inequality, it is easy to construct examples where aggregate utility strictly increases as more and more paths are provided.*

The above result suggests that we would like to provide each user with as large a set of paths possible to perform active load balancing on. However, this comes with the overhead of having to maintain these paths. We examine this issue next by considering a simple policy where a session is given a set of possible paths to draw from, say $\mathcal{R}(s)$ for a class s session, and allows the session to actively use a small subset of these paths, say two of them, while at the same time constantly trying out new paths and replacing poorly performing paths in the old set with better performing paths in the new set. More specifically we consider the following path selection mechanism. Assume a user is using path set c. This user is offered a new path set c' at some fixed rate $A_{cc'}$. This new path set is accepted under the condition that the user receives a higher aggregate rate than it was receiving under c. This process then repeats.

We use the model of Section 2, where the class s users, N_s in number, are divided according to the set of paths they are currently using, $N_c(t)$ denoting the number of class s-users actively using paths in $c \subset \mathcal{R}(s)$ at time t. Class s users actively using the set c of paths consider replacing their path set c with path set c' according to a Poisson process with intensity $A_{cc'}$. We shall assume that $|c| = |c'| = b$, i.e., the number of paths in an active set is fixed at b. Finally, assume that for each class s, any $r \in \mathcal{R}(s)$, any given set $c \in \mathcal{C}(s)$, there is some c' such that $r \in c'$ and $A_{cc'}$ is positive (recall that $\mathcal{C}(s)$ is defined as the collection of size b subsets of $\mathcal{R}(s)$) . This assumption states that all paths available to a class s session should be tried no matter what set of initial paths is given to that session.

We also have to concern ourselves with the sending rates of the different users as path reselection proceeds over time. Let $\lambda_c(t)$ denote the data transfer rate for a user actively using path set c, $\lambda_c(t) = \sum_{r \in c} \lambda_{c,r}(t)$ where $\lambda_{c,r}(t)$ is the sending rate along path r at time t. We have described in [11] a dynamic process where the vectors $\{N_c(t), \lambda_{c,r}(t)\}$ change over time. This process is stochastic in nature and consequently difficult to model. However, if we assume that the population of users in each class is large, which is reasonable for the Internet, then we can model this process over time by a set of ordinary differential equations, representing the path reselection and rate adaptation dynamics of users over their active path sets. Under the condition that the utility functions and penalty functions are well behaved, we can show that $N_c(t)$ converges to a limit N_c and $\lambda_{cr}(t)$ converges to λ_{cr} as t tends to infinity. Remarkably, we can show that these limits are the maximizers of

$$\mathcal{W}(\lambda, N) := \sum_{s \in \mathcal{S}} \sum_{c \subset \mathcal{R}(s)} N_c U_s(\lambda_c) - \Gamma(\Lambda_r) \qquad (10)$$

subject to $\sum_{c \in \mathcal{C}(s)} N_c = N_s$. In other words, this resampling process allows the system to converge to a state where the proportion of class s sessions using active path set $c \in \mathcal{R}(s)$ and the aggregate rates at which they use these paths maximize the aggregate sum of utilities. This is more precisely stated in the following theorem.

THEOREM 5.1. *Assume that the utility functions U_s and the penalty function Γ are continuously differentiable on their domain, that the former are strictly concave increasing, and the latter convex increasing. Assume further that $U_s'(x) \to 0$ as $x \to \infty$. Then $(N_c, \lambda_{c,r})$ converges to the set of maximizers of the welfare function (10) under the constraints $\sum_{c \in \mathcal{C}(s)} N_c = N_s$. The corresponding equilibrium rates (λ_r) are solutions of the coordinated welfare maximization problem (2).*

The proof proceeds by showing that trajectories of the limiting ordinary differential equation are bounded, that welfare increases over time, and then using Lasalle's invariance theorem to prove that the limiting points of these dynamics coincide with equilibrium points of the ordinary differential equation; showing that the equilibrium points coincide with the maximum of (10) completes the proof.

What makes this result especially useful is that benefit maximization on the part of a user conforms *exactly* to the user trying to maximize its rate through the path reselection process. Thus, this path reselection policy is easy to implement: at random times the session initiates data transfer using the coordinated rate controller over a new set of paths and measures the achieved throughput dropping either the old path set or new path set depending on which achieves lower throughput. This equivalence is a consequence of the assumption that the utility U is strictly concave and continuously differentiable.

5.2 Uncoordinated congestion control

As one might expect by now, the story is not as clean in the case of uncoordinated control, and no monotonicity result exists. Indeed, for a symmetric triangle network described in [11], with three source-destination session types, allowing each session to use the two-link path as well as the direct path *decreases* throughput. However random resampling is still beneficial provided that the uncoordinated control exhibits no RTT bias. If a session is given a set of paths to draw from, then the random resampling strategy described earlier maximizes welfare without the need to use all paths. Moreover, it suffices for sessions to use a greedy rate optimization strategy to determine which set of paths to keep in order to ensure welfare maximization. The reader is referred to [11] for further details.

6. DISCUSSION AND DEPLOYMENT

Till now, we have focused on networks supporting workloads consisting of *persistent or infinite backlog flows*. Moreover, the emphasis has been on the effect that multipath has on aggregate utility. In this section we consider workloads consisting of finite length flows that arrive randomly to the network. Our metric will be the capacity of the network to handle such flows. We will observe that several results from previous sections have their counterparts we focus on finite flows.

As before, we represent a network as a capacitated undirected graph $G = (V, E, C)$ supporting a finite set of *flow classes*, $\mathcal{S}$ with attendant sets of paths $\{\mathcal{R}(s)\}$. We assume that class s sessions arrive at rate α_s according to a Poisson process and that they introduce independently and identical exponentially distributed workloads with a mean number of bits $1/\mu_s$. We introduce the notion of a *capacity region* for this network, namely the sets of $\{\alpha_s\}$ and $\{\mu_s\}$ for which there exists some rate allocation over the paths available to the sessions such that the time required for sessions to complete their downloads are finite.

In the case of coordinated control, it is possible to derive the following monotonicity result with respect to the capacity region of the network. Consider a network G that supports a set S of flow classes with arrival rates $\{\alpha_s\}$ and loads $\{\mu_s\}$. Let $\{\mathcal{R}(s)\}$ and $\{\mathcal{R}'(s)\}$ be two collections of paths for these classes that satisfies $\mathcal{R}(s) \subseteq \mathcal{R}'(s)$ for each $s \in S$ and suppose that each session applies coordinated rate and path control over these paths. Then if $\{\alpha_s\}$, $\{\mu_s\}$, lie within the capacity region of the network with path sets $\{\mathcal{R}(s)\}$, they lie in the capacity region of the network with path sets $\{\mathcal{R}'(s)\}$ as well.

REMARK 6.1. *It is easy to find examples where the capacity region strictly increases with the addition of morel paths.*

REMARK 6.2. *Although this result is stated for the case of exponentially distributed workloads, it is straightforward to show that it holds for any workload whose distribution is characterized by a decreasing failure rate. This includes heavy-tailed distributions such as Pareto.*

It is interesting to ask the same question about the capacity region when uncoordinated control is used by all flows. Unfortunately, similar to the infinite session workload case, no such monotonicity property exists.

It is also interesting to ask the question as to which controller yields the larger capacity region. As in the case for finite flows, we can show that for a given network configuration ($G, \mathcal{S}$ and $\mathcal{R}$ fixed), if $\{\alpha_s : s \in \mathcal{S}\}$, $\{\mu_s : s \in \mathcal{S}\}$ lies within the capacity region of the network when operating with an uncoordinated control, then they lie within the capacity region of the network when operating under coordinated control as well.

REMARK 6.3. *It is easy to construct cases where the converse is not true. For instance, the symmetric triangle with single and two-link routing mentioned for fixed flows is such an example (see [12]).*

We conclude from this monotonicity property for coordinated control that more is better. However, improved capacity comes at the cost of increased complexity at the end host, namely maintenance of state for each path and executing rate controllers over each path. Fortunately, as in the case of infinite backlogged sessions, this is not necessary. It suffices for a session to maintain a small set of paths, say two paths, and continually try out random paths from the set of paths available to it, and drop the path which provides it with the poorest performance, say throughput. Note the similarity of this process to that of BitTorrent, which periodically drops the connection providing the lowest throughput and replacing it with a random new connection. Interestingly enough, this multipath algorithm coupled with random resampling achieves the same capacity region as one that requires flows to utilize all paths. Indeed, we can prove the analogy of 5.1:

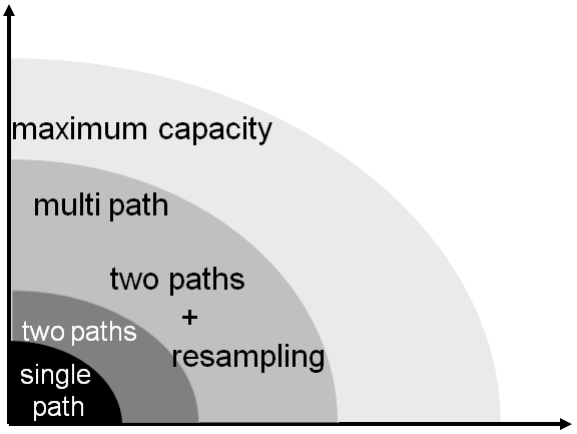

Figure 3: Capacity region under multipath with and without resampling.

THEOREM 6.1. *Assume that class s sessions use all paths from $\mathcal{R}(s)$. Assume the set of loads $\{\alpha_s\}$ and $\{\mu_s\}$ lies within the network capacity region. Consider an approach where a class s session uses a subset of paths from $\mathcal{R}(s)$, randomly samples a new path set according to a Poisson process with rate γ_s and drops the worst of the two path sets. Then $\{\alpha_k\}$ and $\{\mu_k\}$ also lie within the capacity region when flows use this resampling approach in the limit as $\gamma_s \to \infty$.*

Figure 6 illustrates and summarizes our capacity results.

As before it is also interesting to ask about the effect of uncoordinated control coupled with random sampling on capacity. Surprisingly enough, uncoordinated control on a small set of paths coupled with random resampling can often increase capacity over uncoordinated control over the entire set of paths.

6.1 Deployment

To effectively deploy multipath, key ingredients are first, diversity, which is achieved through a combination of multi-homing and random path sampling, and second, path selection and multipath streaming using a congestion controller that actively streams along the

best paths from a working set. Although home-users are currently often limited in their choice of ISP and hence cannot multihome, in contrast campus or corporate nodes often have diverse connections, via different ISPs or through 3G wireless and wired connectivity. Moreover the growth of wireless hotspots, wireless mesh and broadband wireless in certain parts of the globe means that even home users may become multihomed in the future. Recent figures [1] suggest that 60% of stub-ASes (those which do not transit traffic) are multihomed, and [5] claims that with IPv6 type multihoming there are at least two disjoint paths between such stub-ASes.

The multipath controllers we have outlined need to be put into practice. Some high-level algorithms designs are considered in [10] and [7], and practical questions are addressed in [18]. Translating from algorithms derived from fluid models to practical packet-based implementations does require care, however we believe this to be perfectly feasible in practice. Indeed, the IETF has a current Multipath TCP working group, which is looking into adding multipath into TCP.

7. SUMMARY

There are potentially significant gains from combining multipath routing with congestion control. Two different flavors of control are possible: one which coordinates transfers across the multiple paths; and another uncoordinated control with sets up parallel connections. The uncoordinated approach is simpler to implement, however it can suffer from poorer performance while coordinated control is better performing and intrinsically 'fairer'. We have contrasted the two types of control, and shown that with fixed path choices uncoordinated control can yield inferior performance, halving throughput in one example.

If path-choices are allowed to be chosen optimally or 'selfishly' by the end-system, then coordinated control reaches the best system-wide optimum; as indeed does uncoordinated control, but only if the control objective is the same for all paths (unlike current TCP), and also only if all users agree to use the the same number of parallel paths (connections). This optimum can also be reached by limiting each session to a small number of path choices (e.g. 2) but allowing paths to be resampled and better paths to replace existing ones.

This suggests that good design choices for multipath controllers are coordinated controllers or uncoordinated controllers with the RTT bias removed.

Acknowledgment

This work was supported in part by the NSF under award CNS-0519922.

8. REFERENCES

[1] S. Agarwal, C. Chuah, and R. Katz. Opca: Robust interdomain policy routing and traffic control. In *Proc. IEEE Openarch*, April 2003.

[2] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Rao. Improving web availiability for clients with monet. In *Proc. NSDI 2005*, July 2005.

[3] D. Bertsekas and R. Gallager. *Data Networks*. Longman Higher Education, 1992.

[4] B. Cohen. Incentives built robustness in BitTorrent. In *Proceeding of P2P Economics workshop*, June 2003.

[5] C. de Launois, B. Quoitin, and O. Bonaventure. Leveraging network performance with ipv6 multihoming and multiple provider-dependent aggregatable prefixes. *Computer Networks*, 50(8):1145 – 1157, 2006.

[6] K. Gummadi, H. Madhyastha, S. Gribble, H. Levy, and D. Wetherall. Improving the reliability of internet paths with one-hop source routing. In *Proc. 6th OSDI*, December 2004.

[7] H. Han, S. Shakkottai, C. Hollot, R. Srikant, and D. Towsley. Multi-path TCP: a joint congestion control and routing scheme to exploit path diversity in the internet. *IEEE/ACM Trans. on Networking*, 14(6):1260–1271, December 2006.

[8] F. Kelly, A. Maulloo, and D. Tan. communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49:237–252, 1998.

[9] F. P. Kelly. Mathematical modelling of the internet. In B. Engquist and W. Schmid, editors, *Mathematics Unlimited - 2001 and Beyond*, pages 685–702. Springer-Verlag, 2001.

[10] F. P. Kelly and T. Voice. Stability of end-to-end algorithms for joint routing and rate control. *ACM SIGCOMM Computer Communication Review*, 35(2):5–12, 2005.

[11] P. Key, L. Massoulié, , and D. Towsley. Path selection and multipath congestion control. In *INFOCOM07*, May 2007.

[12] P. Key and L. Massoulié. Fluid models of integrated traffic and multipath routing. *Queueing Systems*, 53(1):85–98, June 2006.

[13] P. Key, L. Massoulié, and D. Towsley. Multipath routing, congestion control and load balancing. In *ICASSP 2007*, April 2007.

[14] M. Kodialam, T. Lakshman, and S. Sengupta. Efficient and robust routing of highly variable traffic. In *HotNets*, 2004.

[15] S. Kunniyur and R. Srikant. End-to-end congestion control schemes: Utility functions, random losses and ECN marks. In *INFOCOM 2000*, 2000.

[16] M. Mitzenmacher, A. Richa, and R. Sitaraman. The power of two random choices: A survey of the techniques and results. In P. Pardalos, S. Rajasekaran, , and J. Rolim, editors, *Handbook of Randomized Computing*, pages 255–312. 2001.

[17] J. Mo and J. Walrand. Fair end-to-end window based congestion control. In *SPIE 98, International Symposium on Voice, Video and Data Communications*, 1998.

[18] C. Raiciu, D. Wischik, and M. Handley. Practical congestion control for multipath transport protocols. UCL Technical Report, 2010.

[19] K. Ramakrishnan, S. Floyd, and D. Black. The

addition of explicit congestion notification (ECN) to IP. Technical Report RFC3168, IETF, September 2001.

[20] R. Srikant. *The Mathematics of Internet Congestion Control*. Birkhauser, 2003.

[21] R. Zhang-Shen and N. McKeown. Designing a predictable internet internet network with valiant load-balancing. In *IWQoS*, 2005.