

Multipath Scheduling for Optimal Overlay Data Transfer using Markov Decision Processes

Vinh Bui, Weiping Zhu

The University of New South Wales, Australia

{v.bui, w.zhu}@adfa.edu.au

Abstract—In this paper, we study the problem of optimal data transfer using multiple overlay paths and the goal is to develop a scheduling mechanism to minimize the time required to transfer a given amount of data from a source to a destination. The problem is formulated as a Markov Decision Process (MDP), which selects transmission paths based on their congestion states. A computational effective algorithm namely OPI i.e. Online Policy Iteration is proposed to solve the formulated MDP on the fly. Through simulations using Ns-2, we show superior performance of the proposed mechanism against two classical scheduling schemes, which are Round Robin and Join the Shortest Queue. Since the proposed scheduling mechanism is computationally effective, it is attractive for real-time data transfer.

Index Terms—Path Scheduling, Internet Traffic Control, Markov Decision Processes;

I. INTRODUCTION

Despite a constant improvement in the network infrastructure, applications requiring quality assured data transfers on the best-effort Internet continue to suffer from limited bandwidths, highly varying delays and losses. Initial studies suggest an application can increase its aggregate throughput and reduce end-to-end delays and losses by distributing data over multiple disjoint paths [1], [2], [8], [10], [11], [22]. Nonetheless, how significant the gain in terms of performance the application can achieve, depends on *how multiple paths are chosen* and *how data is distributed among the chosen paths*.

Recently, application-level overlay networks have been used as a feasible yet effective architecture to establish multipath environments [4], [5], [17]. Compared with the efforts to utilize multiple paths at lower levels e.g. in [7], [21], which require the deployment of a new network infrastructure including new multipath capable protocols and supporting hardware, application-level overlay networks have a number of advantages including deployment feasibility, low cost, flexibility and TCP/IP friendliness. Therefore, in this paper, we propose an approach to improve performance of point-to-point data transfers in overlay networks. Specifically, we address the problem of transferring an arbitrary amount of data between two end hosts using multiple overlay paths while minimizing the total completion time. Since the Internet paths exhibit the Markovian property [14], [20], [23], we formulate the problem as a Markov Decision Process (MDP) and solve the formulated MDP using a computational effective algorithm. To the best

of our knowledge, we are the first who tackle this problem under the framework of Markov Decision Processes.

Problems of optimal multipath data transfers on overlay networks have been studied in [8], [10], [22]. In [22], the authors focused on exploiting TCP to increase the aggregate throughput of a point-to-point multipath data transfer. They developed a path rate controller that independently maximizes the sending rate on each path. We, on the other hand, focus on minimizing duration of the transfer by exploiting the path diversity. Unlike the approach presented in [22], our approach is protocol independent i.e. it suits both TCP and UDP.

In a different direction, the authors of [8] and [10] studied the problem of data replication/collection and the objective is to minimize the makespan, which is the duration of the longest transfer. Both studies formulated and solved the problem using graph theory where the graph is the overlay network topology and the constraints are path capacities. Unfortunately, the graph theory approach has a number of potential issues as admitted by the authors of [8]. First, the graph theory formulation assumed the time required to transfer a unit of data through a network link is fixed, which is unrealistic. In present of background traffic, this time is stochastic. Second, since the capacity of a path does not fully correspond to the path end-to-end delays, using path capacities as constraints to minimize the makespan may not lead to an optimal solution. In such a dynamic environment, an approach based on a dynamic optimization framework like ours is more appropriate. Although in this paper we focus on point-to-point data transfers, our approach can be extended to cover point-to-multipoint (replication) and multipoint-to-point (collection).

To highlight the proposed approach, we underline that: (i) it helps to improve the quality of multipath data transfer on overlay networks; (ii) it formulates the minimum completion time data transfer problem over multiple overlay paths as a MDP and solves the formulated MDP using both standard algorithms and a more computational effective one called OPI i.e. Online Policy Iteration; (iii) it confirms superior performance of the proposed approach against classical schemes including RR and JSQ by means of simulation conducted with Ns-2 [18].

The rest of the paper is organized as follows. In Section II and Section III, the problem under study is defined, and the system model is established. The formulation of the problem as a MDP is presented in Section IV. Section V details the algorithm to optimally distribute data over the overlay paths by solving the formulated MDP. In Section VI, the numerical verification of the proposed approach is presented. Finally,

⁰This work is supported by University of New South Wales at Australian Defence Force Academy (UNSW@ADFA).

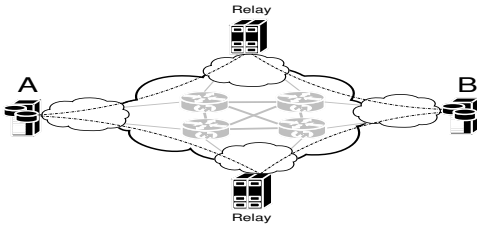


Fig. 1. An application-level overlay network with multiple path support.

concluding remarks are made in Section VII.

II. PROBLEM DEFINITION

A. Problem Setting

Consider an overlay network that supports M , ($M > 1$) paths connecting source A and destination B as illustrated in Fig 1; and a set of data e.g. multimedia contents of variable size to be transferred from source A to destination B.

Without losing of generalization, let us assume that the size of the contents to be transferred is a random variable ν governed by a Geometric distribution with parameter γ :

$$P(\nu = n) = (1 - \gamma)\gamma^{n-1}, (0 \leq \gamma \leq 1); n = 1, 2, \dots \quad (1)$$

The contents are segmented into fixed-size bins i.e. packets before being dumped into a module called *the traffic distributor* with a constant rate λ (the fixed size is assumed to simplify the formulation of the problem, which can be easily generalized for variable size). At the traffic distributor, bins are subsequently schedule to transmit over the M paths.

Our objective is to construct a scheduling policy, which specifies on which overlay path and at what time should each bin of data be transferred to the destination to achieve the minimum transfer time for the whole data set. Since the Internet path conditions i.e. available bandwidths, delays and losses are time-varying, the problem under study belongs to the class of dynamic optimization problems. A powerful framework to tackle this class of problems is Markov Decision Processes [3]. Nonetheless, before the framework can be applied, the underlying system and its states must be clarified.

III. SYSTEM AND STATES

A. System Model

Previous work on similar problems (e.g. on scheduling packets over multiple paths [6] and on streaming video over multiple paths [12]) when applying decision frameworks, often assumes the explicit knowledge of the path/channel loss and delay characteristics. This information is usually difficult to obtain correctly in practice. We avoid this assumption by designing a simple path state monitoring mechanism, which in general can reveal loss and delay states/conditions of the path by keeping track on data bins being transmitted. The mechanism works as follows. Before the transmission of each data bin, the bin ID and its time of transmission are recorded on a list of size K . This information is kept until an acknowledgment of the successful transmission of the bin is received. In this manner, given the data stream, the evolution

of the number of bins on the list will implicitly reflect the path states in terms of Round Trip Time (RTT) delays and losses. The path state monitoring mechanism is illustrated in Fig. 2.

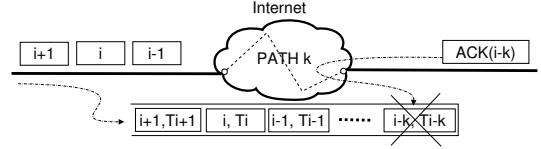


Fig. 2. Path state monitoring mechanism.

It is easy to see by using the path state monitoring mechanism, each path can be modeled as a single server queue. The queue state is the number of bins present in the queue. The arrival process of the queue is dictated by the traffic distributor arrival process and its scheduling policy. The inter-departure intervals between “customers” in the queue are characterized by the distribution of RTT jitters¹. The queue is denoted as $G/G/1/K$ where G/G stands for General arrival time and General service time distributions, 1 means the queue is single server, and K means the buffer length of the queue is K [13].

Since each individual path is modelled as a $G/G/1/K$ queue, the system under study is a set of M parallel queues. The system states are created by vectors of states of all queues in the system. Fig. 3 depicts the system model.

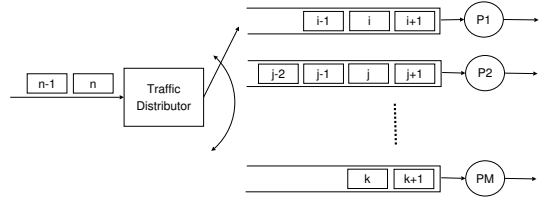


Fig. 3. The system model with M independent paths.

Having defined the system model and its states, in general decisions can be made on how to schedule bins optimally among the paths. However, to apply the MDP framework, we have to identify the Markovian property of the system.

B. Imbedded Markov Chain

At the time instance when a data bin arrives at the traffic distributor, a decision has to be made on where to send the bin. Depending on the decision, the system may change its state in accordance. Therefore, the system behaviors observed at the arrival instances are important for the decision maker. To study these behaviors, the method of Imbedded Markov Chain [13] in combination with renewal theory [9] is applied.

Let t_i be the arrival instance of bin i at the traffic distributor. In accordance, let $Q_m(t_i)$ be the numbers of bins found in the m -th queue at t_i . Consequently, the system state at t_i is a vector $\{Q_m(t_i)\}$, ($m = 1 \dots M$). We are interested in the evolution of $\{Q_m(t_i)\}$, ($i = 1, 2, 3 \dots \infty$) under the impact of the scheduling actions taken at the traffic distributor.

¹RTT jitter is defined as the difference between two consecutive RTT values i.e. $Jitter_i = RTT_{i+1} - RTT_i$.

Assume that at arrival instance t_i , the bin i is sent to the m -th queue by the decision maker. To eliminate the possible transitive period and simplify the discussion, we assume that a decision and its action can be completed instantly. As a consequence, the system will move immediately from state $\{Q_1(t_i), Q_2(t_i), \dots, Q_m(t_i), \dots, Q_M(t_i)\}$ to state $\{Q_1(t_i), Q_2(t_i), \dots, [Q_m(t_i) \leftarrow Q_m(t_i) + 1], \dots, Q_M(t_i)\}$.

From this moment, until the arrival of the next bin, the evolution of $\{Q_m(t_i)\}$ depends only on the number of bins departing from each queue during $[t_{i+1} - t_i]$ interval. In other words, if $P_{QQ'}$ denotes the probability the system is in state $\{Q'_m(t_{i+1})\}$ at the arrival instance of bin $(i+1)$, then $P_{QQ'}$ equals to the probability that there are $Q_1(t_i) - Q'_1(t_{i+1})$ bins departing from the first queue, and $Q_2(t_i) - Q'_2(t_{i+1})$ bins departing from the second queue, and so on during $[t_{i+1} - t_i]$ interval. Since these departure processes are independent, we can obtain $P_{QQ'}$ by taking product of the probabilities.

To compute the probability there are $Q_m(t_i) - Q'_m(t_{i+1})$ bins departing from the m -th queue, we utilize some results of renewal theory. Let τ_k and τ_{k+1} subsequently denote the departure instances of bins k and $k+1$ from the m -th queue. Assuming the inter-departure intervals $[\tau_{k+1} - \tau_k]$ are i.i.d random variables governed by a general distribution $g_m(x) = P(\tau_{k+1} - \tau_k \leq x)$. Clearly, the sequence of departure points $\{\tau_k\}$ fully forms an *ordinary* renewal process.

Suppose the queuing process has been running for a long time and attained a steady state. Subsequently, the sequence of departure points $\{\tau_k\}$ started from the arrival instance t_i forms the *equilibrium* renewal process of the above ordinary renewal process [9]. The illustration is given in Fig. 4.

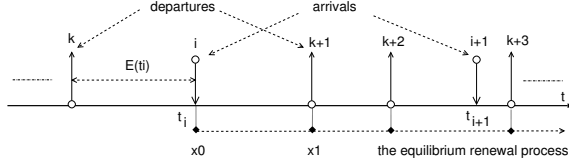


Fig. 4. The queuing process of the m -th queue.

According to renewal theory [9] applying for the *equilibrium* renewal process, the number of bins departing from the queue in $[t_{i+1} - t_i]$ interval is proportional to the length of the interval and does not depend on t_i . Thus, $Q_m(t_{i+1})$ is fully determined by $Q_m(t_i)$ and does not depend on $E(t_i)$, which is the elapsed time since the last departure observed at t_i . The random variable $Q_m(t_i)$ forms a discrete-state Markov chain imbedded in the queuing process. The state space of the chain is comprised of all possible numbers of bins in the queue.

Let β_j denote the probability of j bins, $j = 0 \dots K$, departing from the queue in $[t_{i+1} - t_i]$ interval. Since the sequence of departure points $\{\tau_k\}$ started from t_i forms an equilibrium renewal process, so β_j is the probability of having j renewal during $(0, t]$ interval, where $t = t_{i+1} - t_i$.

Recall that the inter-departure intervals $[\tau_{k+1} - \tau_k]$ are i.i.d random variables with a general p.d.f $g_m(x)$.

Let $G_m(x)$ be the corresponding cumulative distribution function of the inter-departure intervals and $G_m^{(n)}(x)$ be the n -fold convolution of $G_m(x)$.

Let $\mu_m \equiv E[\tau_{k+1} - \tau_k]$ be the common inter-departure length of the queue. Subsequently, as pointed out in [9] the probability of having j renewals during interval $(0, t]$ of the equilibrium renewal process is:

$$\begin{aligned} \beta_j &= P[N(t) = j] \\ &= \frac{1}{\mu} \int_0^t (P[N^o(u) = j-1] - P[N^o(u) = j]) du \\ &= \frac{1}{\mu} \int_0^t [(G_m^{(j-1)}(u) - G_m^{(j)}(u)) - (G_m^{(j)}(u) - G_m^{(j+1)}(u))] du \end{aligned} \quad (1)$$

where $P[N^o(u) = r] = G_m^{(r)}(u) - G_m^{(r+1)}(u)$ is the probability of having r renewals during interval $(0, u)$ of the corresponding *ordinary* renewal process.

As mentioned, the random variable $Q_m(t_i)$ forms a discrete-state Markov chain. Since the state transitions take place only at the arrival epochs, the system state $\{Q_m(t_i)\}, m = 1 \dots M$, also forms a discrete-state Markov chain. Having defined the formula for β_j , the state transition probability matrix of the chain is determined. Fig. 5 illustrates the system chain.

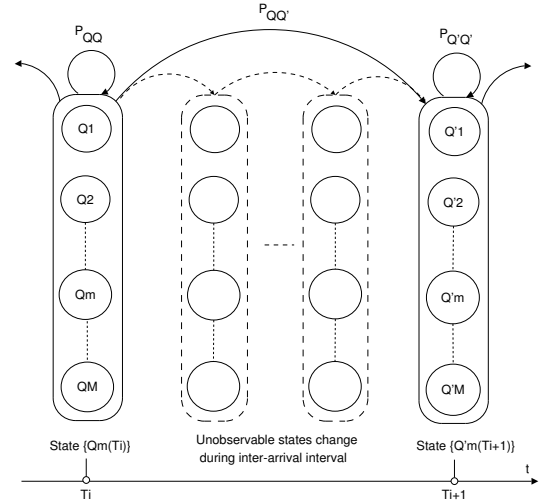


Fig. 5. The system imbedded Markov chain.

C. Transition Probability Matrix Computational Issues

Although we are able to obtain the transition probabilities matrix of the system Markov chain in the general form, it is challenging to compute the matrix with inter-departure intervals following a general distribution $g(x)$. Several approaches are available to cope with the transition matrix computational difficulty including simulation e.g Monte-Carlo, reinforcement learning e.g. Q-learning, and function approximation. Due to the space constraint, we present here only the function approximation approach. In particular, we approximate $g(x)$ by the a -stage Erlang distribution with rate ρ .

We have by approximation:

$$g(x) \approx \frac{\rho^a x^{a-1} e^{-\rho x}}{(a-1)!}$$

Since

$$\int_0^t \frac{\rho^{m+1} u^m e^{-\rho u}}{m!} du = 1 - \sum_{n=0}^m \frac{(\rho t)^n e^{-\rho t}}{n!}$$

We can compute β_j directly:

$$\begin{aligned} \beta_j &= \frac{\rho}{a} \int_0^t \sum_{m=j a-a}^{j a-1} - \sum_{m=j a}^{j a+a-1} \frac{(\rho u)^m e^{-\rho u}}{m!} du \\ &= \frac{1}{a} \sum_{m=j a}^{j a+a-1} \sum_{n=0}^m - \sum_{m=j a-a}^{j a-1} \sum_{n=0}^m \frac{(\rho t)^n e^{-\rho t}}{n!} \\ &= \frac{1}{a} \sum_{n=j a}^{j a+a-1} (j a + a - n) \frac{(\rho t)^n e^{-\rho t}}{n!} \\ &\quad + \frac{1}{a} \sum_{n=j a-a}^{j a-1} (n - j a + a) \frac{(\rho t)^n e^{-\rho t}}{n!} \end{aligned}$$

By assumption, bins arrive evenly at the traffic distributor with a constant rate λ . Subsequently, $t = t_{i+1} - t_i = 1/\lambda$. Substituting to the above β_j formula, we get:

$$\begin{aligned} \beta_j &= \frac{1}{a} \sum_{n=j a}^{j a+a-1} (j a + a - n) \frac{(\rho/\lambda)^n e^{-\rho/\lambda}}{n!} \\ &\quad + \frac{1}{a} \sum_{n=j a-a}^{j a-1} (n - j a + a) \frac{(\rho/\lambda)^n e^{-\rho/\lambda}}{n!} \end{aligned}$$

Given ρ/λ ratio, we can be compute β_j directly from the above formula. Obtaining β_j for each queue we can compute the state transition probability matrix of the Markov chain.

IV. MDP FORMULATION

A. Markov Decision Process

Consider a time homogeneous MDP [19] with a countable state space S , a finite action space A where $A(s) \in A$, $s \in S$ is a set of admissible actions in state s , a non-negative immediate reward function $R : S \times A(S) \rightarrow \mathfrak{R}_+$, and a set of conditional probabilities $P(s'|s, a)$, which is the probability of moving from state s to state s' if action $a, a \in A(s)$ is taken.

In unconstrained MDP formulation, the MDP has a single objective, which is to maximize (minimize) the total (average) rewards achieved over a period of time. In context of the studied problem, the objective is to minimize the transfer delay of the whole data set. Subsequently, the four-component tuple $\{S, A, P, R\}$ of the MDP is made of:

- S is the state space of the defined system consisted of M , ($M \geq 1$) overlay paths. It is clear that S is the state space of the system embedded Markov chain.
- $A = \{a_1, a_2 \dots a_M\}$ is the set of M possible actions each of which corresponds to the action of scheduling a data bin to one of M overlay paths.
- P is the state transition probability matrix of the system Markov chain, which can be calculated from path state transition probabilities as shown in Section III-C.
- R is the immediate reward function, which should be able to reflect the minimum transfer time objective. According to Little's theorem [16], the average delay experienced by

a bin is related to the average number of bins in the queue buffer $\bar{D} = \frac{1}{\lambda} E[C_i]$. Therefore, we define the immediate reward function: $R(s, a) = \mu_a s + d_a$ where μ_a is the common inter-departure interval of the path chosen by action a ; $s \in \{C_1, C_2 \dots C_K\}$ is the state of the path; and d_a is the path propagation delay.

A Markov policy is a description of behaviors, which specifies the action to be taken in correspondence to each system state and time step. If a policy is *stationary*, it specifies only the action to be taken in each state independently from the time steps. Given policy π and initial state s of the system, one can quantitatively evaluate π based on the expected cumulative reward, which is defined as follows:

$$v^\pi(s, T) \equiv E_s^\pi \left\{ \sum_{t=1}^T R(S(t), a_\pi(S(t))) \right\} \quad (2)$$

where $v^\pi(s, T)$ denotes the expected cumulative reward achieved by the decision maker from time step 1 to time step T with initial state $s, s \in S$; $R(S(t), a_\pi(S(t)))$ denotes the immediate expected reward received by the decision maker at time t when taking action $a_\pi(S(t))$ in accordance with the policy π while the system is in state $S(t)$.

Recall that the size of the contents in the data set to be transferred is a random variable ν governed by a Geometric distribution with parameter γ , ($0 \leq \gamma \leq 1$).

$$P(\nu = n) = (1 - \gamma)\gamma^{n-1}, \quad n = 1, 2, \dots$$

Subsequently, the expected cumulative reward obtained by the decision maker when using policy π to transfer the data set can be defined as follows:

$$v^\pi(s) \equiv E_s^\pi \left\{ \sum_{n=1}^{\infty} \sum_{t=1}^n R(S(t), a_\pi(S(t))) (1 - \gamma)\gamma^{n-1} \right\} \quad (3)$$

Since the delay has a finite value, (3) can be further simplified [19], which gives:

$$v^\pi(s) = R(s, a_\pi) + \sum_{s' \in S} \gamma p_a(s'|s) v^\pi(s') \quad (4)$$

where $R(s, a_\pi)$ is the immediate expected reward received when taking action a in state s in accordance to policy π ; and $p_a(s'|s)$ is the probability the system will move from state s to state s' when action a is taken.

An optimal policy is the one that minimizes the cumulative reward v^π . Note that, it is sufficient to find an optimal policy in the Markov policy space since, for any history-dependent policy there is a Markov policy that yields the same cumulative reward. In this MDP formulation, it is possible to find a stationary optimal policy since ($0 < \gamma < 1$) [19].

V. OPTIMAL DATA TRANSFER

A. Scheduling with On-line Policy Iteration

An optimal scheduling policy can be achieved by solving the formulated MDP. To obtain the MDP optimal policies, different approaches can be used e.g. dynamic programming and linear programming. However, we pay more attention on the solving time since the problem under study is time sensitive. In particular, we implement a computational effective

algorithm called On-line Policy Iteration to gradually approach the optimal policy of the formulated MDP on the fly.

On-line Policy Iteration (OPI) is designed on the idea of the Policy Iteration (PI) algorithm [19] and asymmetric dynamic programming. Two key factors, which potentially make OPI a preferable algorithm for time sensitive problems are incremental on-line policy improvement and computational efficiency. The first factor comes naturally since OPI is based on the PI algorithm, which iteratively improves a given random policy until an ϵ -optimal policy is found. The second factor however is resulted from a smart choice of the starting policy and the improvement strategy. Particularly, instead of choosing a random policy for improvement like being done in the PI algorithm, we select the Join the Shortest Queue (JSQ) policy to be the starting policy for OPI. As the JSQ policy is an optimal policy for 2 parallel symmetric queues and an acceptable policy for several other cases [15], we can save a lot of iterations by starting from it on the way to reach an ϵ -optimal policy. The policy improvement strategy also plays an important role in saving computational efforts. It is common in practice when some states of a system are more frequently visited than the others. These states are usually more important to the decision maker as they contribute more to the final outcome. Therefore, instead of equally evaluating and improving the policy for every state as in Policy Iteration algorithm, we could spend more time on evaluating and improving the policy for those important states. This policy improvement strategy is known as prioritized sweeping, which still guarantees an ϵ -optimal policy will be found if all system states are asymptotically visited [19]. The OPI algorithm is:

- **Step 1:** Evaluate JSQ policy to obtain $v^{opi}(s) = v^{jsq}(s)$.
- **Step 2:** Obtain the system current state s . Take action $a \in A$ that minimizes $v^{opi}(s)$ and update $v^{opi}(s)$ in (4) with the maximum value:

$$v^{opi}(s) = \min_{a \in A} \{R(s, a) + \sum_{s' \in S} \gamma p_a(s'|s) v^{opi}(s')\}$$

where $p_a(s'|s)$ denotes the transition probability from state s to state s' if action a is taken.

- **Step 3:** If there is a decision to be made, i.e. more data to send, return to Step 2. Otherwise, stop.

Since the OPI algorithm is a special case of the PI algorithm with policy improvement step being done on the fly, the convergence of the algorithm is proved when $0 < \lambda < 1$. The details of the prove can be found in [19].

B. Complexity Analysis

Since the most computational expensive step of OPI, which is the evaluation of JSQ policy can be done offline immediately whenever the system model is available, the OPI algorithm is lightweight and can be used for making decisions on the fly. In specific, we only have to compute (4), the complexity of which is $O(N)$, ($N = \text{sizeof}(S)$) for each $a \in A$. In practice, N is usually in the range of thousands depending on the buffer size K and the number of transmission paths M .

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed approach by means of simulation using Ns-2 [18]. We implemented 4 scheduling algorithms including RR, JSQ, OPI and MDP, which is an optimal policy solved with the standard algorithms. We subsequently evaluated the performance of the algorithms and study the influence of different factors e.g. the source sending rate, the number of overlay paths as well as the path dependence to the performance in two scenarios: homogeneous and heterogeneous overlay paths. Finally, we discuss about the sensitivity and robustness of the algorithms and propose some directions for improvements. Unless otherwise specified, the results presented are summarized over 30 runs in the form of normalized mean \pm standard deviation.

1) *Simulation Setup:* We setup a simulation network of 10 overlay paths, each overlay path contains 1 relay to guarantee a certain level of the path independence. The simulation network topology is illustrated in Fig. 6(a). Background traffic of each path is generated using a random number of FTP, CBR and ON/OFF traffic sources, the parameters of which e.g. file size, sending rate and on/off time are chosen randomly to maintain the dynamics of the network. Fig. 6(b) shows the variation of the background traffic through the evolution of the bottleneck link queue. Data bin size is 500 bytes, which is the common size of Real Network audio/video packets.

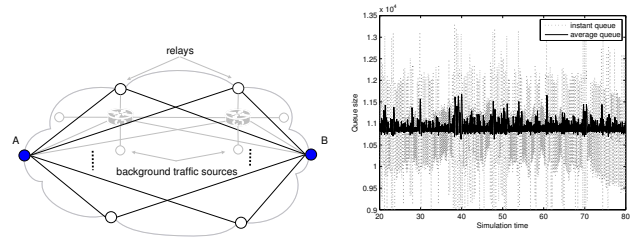


Fig. 6. (a)Simulation network topology. (b)Background traffic dynamics.

2) *Performance Comparison:* We compare the performance of RR, JSQ, OPI and MDP algorithms under different background traffic intensities with 2, 3 and 4 overlay paths. Since the observed results are not significantly different, we only present the result obtained from the case of 3 overlay paths in both homogeneous and heterogeneous paths scenarios.

In the homogeneous paths scenario, each overlay paths has the capacity of approximately 0.1Mbps and the propagation delay of 20ms. The source sending rate is set to 384Kbps. We evaluate the performance of the scheduling algorithms when transferring ν amount of data ($0.1Mb \leq \nu \leq 100Mb$). The performance metric is the transfer time of each algorithm normalized over the transfer time of the RR. The comparison in this scenario is illustrated in Fig. 7(a).

For the heterogeneous path scenario, we setup 3 overlay paths the capacities of which are approximately 0.3Mbps, 0.2Mbps and 0.1Mbps. The propagation delay of the overlay paths are set to 10ms, 15ms and 20ms respectively. All other simulation settings are the same as in the homogeneous paths scenario. The comparison is depicted in Fig. 7(b).

As shown by Fig. 7, in both scenarios, the performance gained by using the OPI algorithm and the MDP optimal

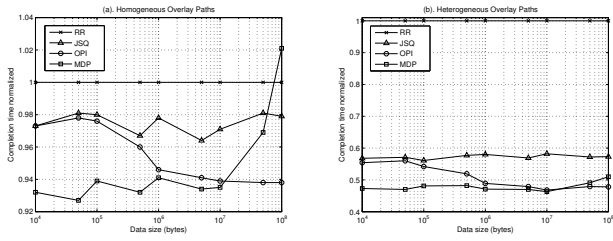


Fig. 7. Performance Comparison of Scheduling Algorithms.

policy is significant. We also can observe the effect of the system nonstationarity, which causes a significant drop in performance of the MDP algorithm when transferring 50Mb and 100Mb data set. Since the path models i.e. ρ/λ ratio used to calculate the MDP optimal policy were estimated only at the beginning of each transfer, they were not capable of reflecting the path characteristics in the long run. As the system model changed, the policy obtained was no longer optimal. As a consequence, the performance of the MDP scheduler was significantly dropped. In contrast, OPI and JSQ could maintain their performance in the long run as their path models had been updated periodically every time 10Mb of data.

In order to evaluate how well the OPI algorithm approaches the optimal policy, we also carried out some KS-tests, which show whether performance of OPI is significantly different from performance of JSQ and MDP. The KS-test results are given in TABLES I, II, III, IV in both homogeneous and heterogeneous paths scenarios. In the tests, the performance metric is the transfer time in $Ns-2$ time unit. $P(H_0)$ is the probability of the test null hypothesis, which assumes that the difference between the test sets is due to chance. D denotes the distance between the test sets.

As shown in TABLES I, III, with the increase of the transfer size, the difference in performance between the OPI and the MDP algorithms becomes not significant as indicated by the increase of the probability $P(H_0)$. On the other hand, the performance of the OPI algorithm becomes significantly better than the performance of the JSQ algorithm when the transfer size reaches 1Mb as shown in TABLE II and 0.5Mb as presented in TABLE IV. The obtained test results also indicate that the OPI algorithm is capable of approaching the optimal policy within the first 10Mb of the transferred data.

3) *Effect of source sending rate:* In order to investigate whether we can improve the performance of the algorithms by increasing the sending rate, we have carried out the experiments in two scenarios. In the first scenario, we utilize 3 homogeneous overlay paths, each of them has the capacity of 0.1Mbps and the propagation delay of 20ms. In the second scenario, we setup 3 heterogeneous overlay paths with 0.3Mbps, 0.2Mbps and 0.1Mbps capacities. The propagation delays of those paths are set to 10ms, 15ms and 20ms respectively. In the both scenarios, we examine the correlation between the sending rate and the performance of RR, JSQ, OPI and MDP schedulers when sending 10Mb data set. Fig. 8(a) illustrates the experiment result in the first scenario. The result of the second scenario is depicted in Fig. 8(b).

As depicted by Fig. 8, in both scenarios, when the sending

TABLE I
OPI VS. MDP WITH HOMOGENEOUS PATHS.

	OPI	MDP	D	$P(H_0)$
0.5Mb	34.95 ± 1.29	33.96 ± 0.74	0.50	0.001
1.0Mb	60.19 ± 1.56	59.79 ± 1.18	0.23	0.34
5.0Mb	276.32 ± 16.4	274.12 ± 9.59	0.13	0.93
10Mb	534.2 ± 13.07	531.7 ± 12.25	0.26	0.20

TABLE II
OPI VS. JSQ WITH HOMOGENEOUS PATHS.

	OPI	JSQ	D	$P(H_0)$
0.1Mb	7.36 ± 0.06	7.37 ± 0.06	0.13	0.93
0.5Mb	34.95 ± 1.29	35.21 ± 1.48	0.1	0.99
1.0Mb	60.19 ± 1.56	62.21 ± 1.65	0.50	0.001
10Mb	534.2 ± 13.1	552.3 ± 14.1	0.56	< 0.001

TABLE III
OPI VS. MDP WITH HETEROGENEOUS PATHS.

	OPI	MDP	D	$P(H_0)$
0.5Mb	14.91 ± 1.82	13.86 ± 1.61	0.30	0.109
1.0Mb	23.38 ± 2.23	22.54 ± 2.15	0.30	0.109
5.0Mb	123.72 ± 7.35	121.46 ± 6.96	0.26	0.20
10Mb	247.9 ± 10.43	245.2 ± 10.24	0.23	0.34

TABLE IV
OPI VS. JSQ WITH HETEROGENEOUS PATHS.

	OPI	JSQ	D	$P(H_0)$
0.01Mb	0.41 ± 0.02	0.42 ± 0.02	0.16	0.76
0.05Mb	1.49 ± 0.06	1.52 ± 0.06	0.30	0.109
0.5Mb	14.91 ± 1.82	16.57 ± 2.02	0.33	0.055
1.0Mb	23.38 ± 2.23	27.72 ± 2.30	0.33	< 0.001

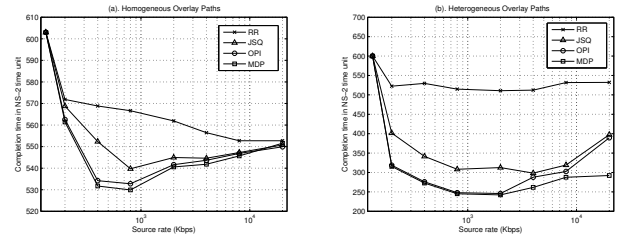


Fig. 8. Effect of source sending rate on performance.

rate is low, the performance of the algorithms increases alongside the increase of the sending rate. However, when the sending rate is high enough, further increase of the sending rate does not improve the performance. In contrast, the increase of the sending rate in the second scenario even leads to the decrease in performance of the JSQ, OPI and MDP algorithms. The reason is that by increasing the sending rate, we will reach to the point when the buffer of the path state monitoring mechanism of each overlay path is saturated. As a result, any decision made under this circumstance is blind since only one system state is observed, which is the state of buffer full. If we continue to increase the sending rate, the performance of the algorithms will converge to a fixed value, which is the performance of the RR algorithm.

4) *Effect of the number of overlay paths:* We verify the statement of [22] claiming that most of performance gains can be realized by using 2 to 4 overlay paths with small extra amount of bandwidth by setting up a simulation of sending 10Mb of data over subsequently 2 to 10 homogeneous overlay paths with a fixed sending rate of 384Kbps. The capacity of

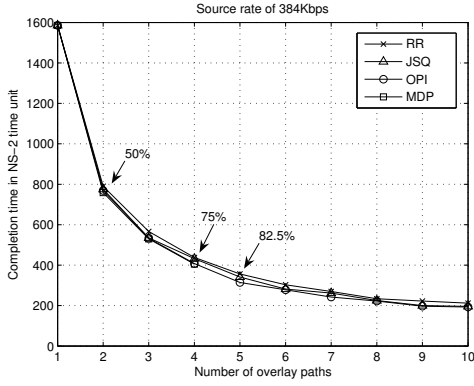


Fig. 9. Effect of the number of overlay paths on performance.

each overlay path is 0.1Mbps and the propagation delay is 20ms. The simulation result is depicted in Fig. 9.

As depicted by Fig. 9, subsequently 50% and 75% of performance gains are realized with 2 and 4 overlay paths. A further increase of the number of paths does increase the performance gains. Nonetheless, the increase is not as significant as before. Thus, we conclude that most of the performance gains will be realized with 2 to 4 overlay paths.

5) *Effect of path dependence*: So far we have setup the simulations based on the assumption of the independence of the overlay paths. However, in practice, it is quite common for some overlay paths to share a physical link, which makes the paths more or less depend on each other i.e. traffic on one path could affect the characteristics of the other paths. As a result, the algorithms like OPI and MDP, which assume the independence of the overlay paths, could suffer some performance losses. In order to investigate the effect of the path dependence on the performance of those schedulers, we have carried out the following simulation.

We create a simulation network consists of 3 overlay paths sharing a 0.3Mbps link. The propagation delay of each paths is subsequently set to 20ms. We then evaluate the effect of path dependence by obtaining the performance of the schedulers when transferring 10Mb data set over the above overlay paths using different sending rates. The obtained performance is then compared with the performance of the schedulers when transferring data over 3 independent homogeneous overlay paths with equivalent capacity and propagation delay. The simulation results is illustrated in Fig. 10, which shows the percentage of performance losses of the schedulers due to the effect with the increase of the source sending rate.

As shown by Fig. 10 the dependence of the overlay paths has a negative impact on the overall performance of the schedulers. In our simulation, the performance loss due to this effect is approximately 6% in average. Compared to the RR algorithm, the OPI and MDP algorithms experienced more losses in performance since the assumption on independence of the overlay paths did not hold. As the sending rate increased, the shared link became more saturated. As a result, traffic on one path had more impact on the other paths. The performance losses suffered by the OPI and MDP algorithms, therefore, became greater.

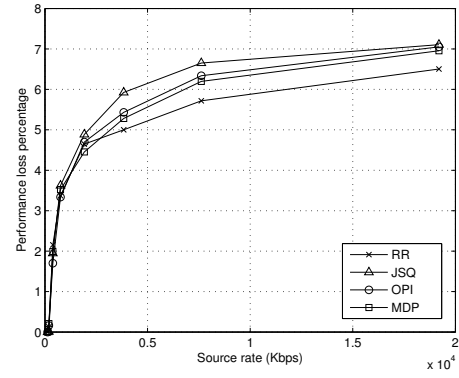


Fig. 10. Effect of the paths dependence on performance.

TABLE V
LOAD SHARE BETWEEN HOMOGENEOUS OVERLAY PATHS.

No Path	2 paths		3 paths		
Delay %	0.50	0.50	0.33	0.33	0.33
RR	50.00%	50.00%	33.33%	33.33%	33.33%
JSQ	48.71%	51.29%	32.71%	33.05%	34.24%
OPI	49.68%	50.42%	32.82%	33.51%	33.67%
MDP	49.52%	50.48%	33.17%	33.38%	33.45%

TABLE VI
LOAD SHARE BETWEEN NONHOMOGENEOUS OVERLAY PATHS.

No Path	2 paths		3 paths		
Delay %	0.25	0.75	0.50	0.30	0.20
RR	50.00%	50.00%	33.33%	33.33%	33.33%
JSQ	85.55%	14.45%	11.73%	24.11%	64.16%
OPI	76.89%	23.11%	16.81%	23.68%	59.51%
MDP	76.08%	23.92%	17.16%	22.73%	60.11%

6) *Load Share Analysis*: We found that, in the long-term, the optimal policy could achieve a load share reversely proportional to the mean delay of each overlay path. If we have M , ($M > 1$) overlay paths and the mean delay of each overlay path is denoted as $\hat{d}_1 \dots \hat{d}_M$ respectively then the portion of load share l_i of path i in the long term is:

$$l_i = \frac{\prod_{j=1, j \neq i}^M \hat{d}_j}{\sum_{i=1}^M \prod_{j=1, j \neq i}^M \hat{d}_j}$$

TABLES V, VI show the percentage of load share on each overlay paths obtained by the algorithms when transferring 10Mb data set. As shown in the homogeneous paths scenario, there is no significant difference in percentage of load share between the algorithms as they all work in a near optimal point. However, in the scenario of nonhomogeneous overlay paths, the difference in load share among the overlay paths is significant between the algorithms. While the MDP and OPI algorithms could maintain a fair share, the JSQ and RR could not as they worked relatively far from the optimal point.

7) *Sensitivity and Robustness of the Algorithm*: Recall the formula used to compute β_j given in section III-C, we can see OPI is only affected by ρ/λ ratio but not the absolute values of ρ and λ . Thus, as far as the ratio is maintained, the OPI performance is guaranteed. Moreover, OPI can resist to a certain level of the variation. Fig.11 shows the values of β_j computed using different ρ/λ ratio. We can see, β_j

values computed with ρ/λ ranging from 1.0–1.5 are relatively closed, which consequently would give a similar performance.

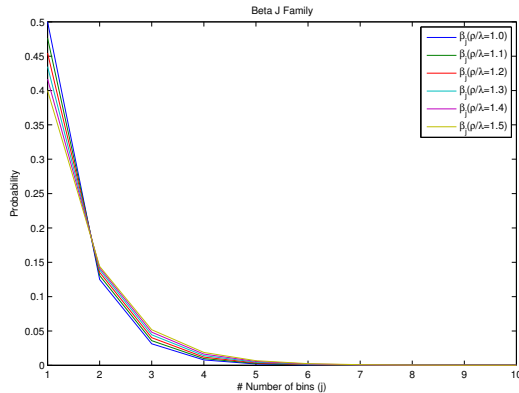


Fig. 11. β_j Family.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have considered the problem of multiple paths scheduling with a minimum completion time objective in the context of multiple paths data transfer using application level overlay networks. In particular, we have formulated the problem as a MDP and implemented an algorithm namely OPI to solve the MDP on the fly. We have evaluated the performance of the proposed algorithm using simulation and have showed that the performance gains obtained by using the proposed algorithm is significant under different network conditions. We also carried out investigation on the effect of different factors e.g. the source sending rate, the number of overlay paths and the path dependence on the performance of the algorithm. We observed that the performance gains will be realized with 2 to 4 overlay paths. In future work, we will investigate the effect of data reordering when using multiple paths and propose solutions to minimize this effect. The work on data reordering will help to bring our solution to solve the problem of optimal multiple path audio and video streaming. To enhance the robustness of the algorithm, work on applying Risk-Sensitive MDP will also be carried out.

REFERENCES

- [1] A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman, "A measurement-based analysis of multihoming," in *Proc. of the ACM SIGCOMM 2003*, Karlsruhe, Germany, Aug. 2003, pp. 353–364.
- [2] A. Akella, J. Pang, B. Maggs, S. Seshan, and A. Shaikh, "A comparison of overlay routing and multihoming route control," in *Proc. of the ACM SIGCOMM 2004*, USA, Aug. 2004, pp. 93–106.
- [3] E. Altman, "Applications of markov decision processes in communication networks," in *Handbook of Markov Decision Processes: Methods and Applications*, E. A. Feinberg and A. Shwartz, Eds. Norwell, Massachusetts, USA: Kluwer, 2002.
- [4] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," *ACM SIGOPS Operating Systems Review*, vol. 35, no. 5, pp. 131–145, Dec. 2001.
- [5] D. G. Andersen, "Improving end-to-end availability using overlay networks," Ph.D., Massachusetts Institute of Technology, Feb. 2005.
- [6] C. Cetinkaya and E. W. Knightly, "Opportunistic traffic scheduling over multiple network paths," in *Proc. IEEE INFOCOM*, vol. 3, Hong Kong, Mar. 2004, pp. 1928–1937.

- [7] J. Chen, P. Druschel, and D. Subramanian, "An efficient multipath forwarding method," in *Proc. IEEE INFOCOM*, vol. 3, USA, Mar. 1998, pp. 1418–1425.
- [8] W. C. Cheng, C.-F. Chou, L. Golubchik, S. Khuller, and Y.-C. Wan, "Large-scale data collection: a coordinated approach," in *Proc. IEEE INFOCOM*, vol. 1, USA, Mar. 2003, pp. 218–228.
- [9] D. R. Cox, *Renewal Theory*. London, GB: Methuen & Co. Ltd, 1962.
- [10] S. Ganguly, A. Saxena, S. Bhatnagar, S. Banerjee, and R. Izmailov, "Fast replication in content distribution overlays," in *Proc. IEEE INFOCOM*, vol. 4, USA, Mar. 2005, pp. 2246–2256.
- [11] J. R. Iyengar, P. D. Amer, and R. Stewart, "Concurrent multipath transfer using transport layer multihoming: performance under varying bandwidth proportions," in *Proc. IEEE MILCOM*, vol. 1, Monterey, CA, USA, Oct. 2004, pp. 238–244.
- [12] D. Jurca and P. Frossard, "Video packet selection and scheduling for multipath streaming," *IEEE Trans. Multimedia*, vol. 9, no. 3, pp. 629–641, Apr. 2007.
- [13] L. Kleinrock, *Queueing Systems Volume 1: Theory*. New York, USA: John Wiley & Son, 1975.
- [14] A. Konrad and A. D. Joseph, "Choosing an accurate network path model," in *Proc. ACM SIGMETRICS 2003*, USA, June 2003, pp. 314–315.
- [15] J. Y.-T. Leung, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. USA: Chapman and Hall/CRC, 2004.
- [16] J. Little, "A proof of the queueing formula $l = \lambda w$," *Operations Research*, vol. 9, pp. 383–387, 1961.
- [17] Z. Ma, H. R. Shao, and C. Shen, "A new multi-path selection scheme for video streaming on overlay networks," in *Proc. IEEE International Conference on Communication*, vol. 3, Paris, France, June 2004, pp. 1330–1334.
- [18] NS-2. (2006, July) Network simulator 2. [Online]. Available: <http://nslam.isi.edu/nslam/index.php>
- [19] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. USA: John Wiley and Sons, 1994.
- [20] K. Salamati and S. Vaton, "Hidden markov modeling for network communication channels," in *Proc. ACM SIGMETRICS 2001*, USA, June 2001, pp. 92–101.
- [21] S. Vutukury and J. J. Garcia-Luna-Aceves, "Mpath: A loop-free multipath routing algorithm," *Elsevier Journal of Microprocessors and Microsystems*, vol. 24, pp. 319–327, 2000.
- [22] B. Wang, J. Kurose, D. Towsley, and W. Wei, "Multipath overlay data transfer," University of Massachusetts Amherst, Amherst, MA, USA, Tech. Rep. CMPSCI-TR05-45, 2005.
- [23] W. Wei, B. Wang, and D. Towsley, "Continuous-time hidden markov models for network performance evaluation," *Elsevier Performance Evaluation*, vol. 49, no. 1-4, pp. 129–146, Sept. 2002.