This paper was presented as part of the main technical program at IEEE INFOCOM 2011

Multi-Path Routing and Rate Allocation for Multi-Source Video On-Demand Streaming in Wireless Mesh Networks

Yong Ding, Yang Yang, Li Xiao
Department of Computer Science and Engineering
Michigan State University
Email: {dingyong, yangyan5, lxiao}@cse.msu.edu

***Abstract*—We study the multi-source video on-demand application in multi-channel multi-radio wireless mesh networks. When a user initiates a new video request, the application can stream the video not only from the media servers, but also from the peers that have buffered the video. The multi-path multi-source video on-demand streaming has been applied in wired networks with great success. However, it remains a challenging task in wireless networks due to wireless interference. In this paper, we first focus on the problem of finding the maximum number of high-quality and independent paths from the user to the servers or peers for each VoD request by considering the effect of wireless interference. We formulate it as a constrained maximum independent paths problem, and propose two efficient heuristic path discovery algorithms. Based on the multiple paths discovered, we further propose a joint routing and rate allocation algorithm, which minimizes the network congestion caused by the new VoD session. The algorithm is aware of the optimization for both existing and potential VoD sessions in the wireless mesh network. We evaluate our algorithms with real video traces. Simulation results demonstrate that our algorithm not only improves the average video streaming performance over all the coexisting VoD sessions in the network, but also increases the network's capacity of satisfying more subsequent VoD requests.**

I. Introduction

The video on-demand application (VoD) has become a popular Internet service recently. There have already been several commercial products developed to support VoD applications, such as PPLive and PPStream. Most of them use peer-to-peer ($P2P$) technology to improve the VoD performance. Such architecture has been discussed in [1] [2]. Assume users can store the videos that they have recently watched in their local storage (e.g., PPLive and PPStream buffers $1G$ bytes of the most recently watched videos, which is enough for over two 2-hour movies, in a peer's local storage). When a client wants to watch a new video, he or she first discovers which peer clients have buffered the video, and then streams the video from both the servers and peer clients through multiple paths. The multi-path multi-source video on-demand streaming has been applied in wired networks with great success. However, it remains a challenging task in wireless networks due to the effect of wireless interference.

As wireless mesh networking technology is attracting more interest in research and industry, it is envisioned to be used for low-cost infrastructures for last-mile Internet access and building community networks [3]. A community network is a static multi-hop wireless network composed of many mesh routers, where each mesh router establishes connectivity with neighboring mesh routers. There are some special routers working as gateways to provide access to the Internet. In a large community network, when a VoD user initiates a video request, it can stream the video from two types of sources: 1) The servers or peers that have buffered the video within the community network; 2) Even if there are no such sources, the user can stream the video from multiple servers or peers in the Internet through the multiple gateways. As VoD applications have high demand of bit rate, delay, and loss sensitivity, the bottleneck of the multi-path video streaming is usually in the community network, that is, the multi-hop wireless paths from the user to peers or servers in the community network or the paths from the user to gateways. Fortunately, in multi-channel multi-radio wireless mesh networks, the enhanced channel diversity increases the network capacity. In this paper, we study multi-path routing and rate allocation in multi-channel multi-radio wireless mesh networks to improve the VoD performance.

One major problem for the multi-source VoD application in wireless mesh networks is the discovery of multiple independent paths. By splitting the video stream over multiple independent paths, we can not only improve the aggregate routing performance, but also improve the stability and robustness. In the Internet, the independent paths are usually defined as edge-disjoint or vertex-disjoint paths. In edge-disjoint paths, no two paths share a same link, and therefore any link failure will only affect one path. Vertex-disjointness is stronger than edge disjointness, because it also guarantees that any node failure will affect at most one path. In wireless mesh networks, the discovery of independent paths becomes more challenging due to wireless interference. Even if two paths are edge-disjoint or vertex-disjoint, they might still affect each other if they have wireless links that interfere with each other. Thus, their aggregate routing performance becomes lower than expected, and the congestion of one path will probably influence the other path. This route coupling effect has been studied in [4]. Therefore, in order to find independent paths in wireless mesh networks, we should guarantee interference-disjointness in addition to edge-disjointness or vertex-disjointness.

Another challenge is that the optimization of VoD performance should be considered over all VoD users in the network instead of only the current user. As we know, the wireless mesh network is designed to be shared among multiple users. If the routes with required bandwidth have been found for a VoD request, the VoD user can establish connections in the network for data transport, which we call a VoD session. There might be multiple coexisting VoD sessions from different users in the wireless mesh network. Thus, we should not be too selfish when finding the routes and rate allocation for the current VoD request. Instead, we should consider not only the resulting performance of the current VoD session, but also its influence on the other existing VoD sessions in the network and the network's ability of satisfying new VoD requests.

Previous studies of multi-path discovery in wireless ad hoc networks either find only edge-disjoint or vertex-disjoint paths without considering wireless interference [5] [6] [7] [8], or focus on multi-path single-source video streaming (find multiple paths between the single video source and the VoD user) [9] [10]. The CAM metric [11] has been proposed for multipath routing with minimum interference in wireless mesh networks, but it is also limited to a pair of single source and single destination. An interference-load aware routing algorithm has been proposed in [12] to route traffic through congestion free areas, but it is limited to single path only. There are only few studies of multi-path multi-source video streaming in wireless mesh networks [13] [14]. However, they focus on single-channel wireless mesh networks and do not guarantee the interference disjointness among the multiple paths. The rate allocation of multi-path video streaming has been discussed in [15] [16] [17]. These algorithms determine the sending rate on each path to optimize the streaming performance of the current VoD session. However, their optimization does not consider the coexisting and potential VoD sessions in the network.

The contribution of this paper is in the following aspects: (1) We consider the effect of wireless interference in the independency of the multiple paths used for video streaming in multi-channel multi-radio wireless mesh networks. We propose two heuristic algorithms to discover the maximum number of independent and high-quality paths that can be used for multi-path video streaming. (2) Based on the multiple paths discovered, we further propose a joint multi-path routing and rate allocation algorithm with the goal of minimizing the network congestion. The proposed algorithm determines the routes together with the rate allocated for each route for the VoD request. By using this optimization framework, we not only improve the average performance of existing VoD sessions, but also enable the network to support more subsequent VoD requests. (3) We use the video trace simulations, and evaluate our algorithms by both network layer metrics and application layer metrics.

The rest of paper is organized as follows. In Section II, we present the system model and problem formulation. In Section III, we describe two proposed heuristic algorithms for multi-path discovery. In Section IV, we propose a joint routing and rate allocation algorithm. The simulation methodology and results are shown in Section V and Section VI, and we conclude our work in Section VII.

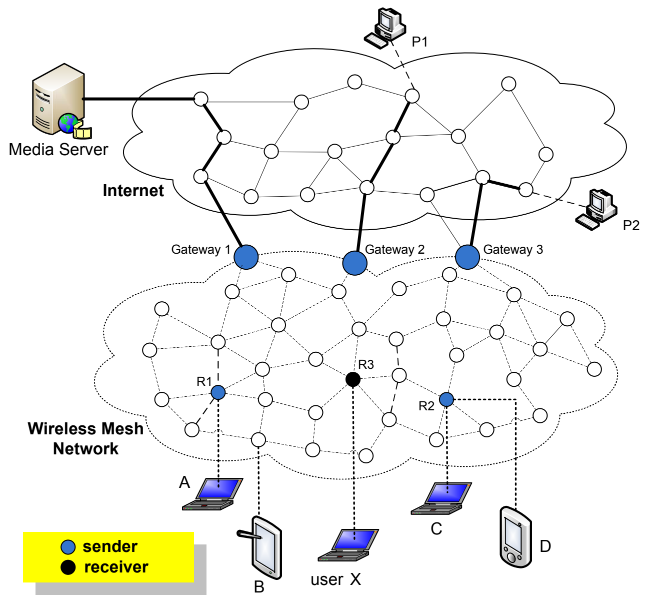

Fig. 1: VoD in Wireless Mesh Networks

II. System Model and Problem Formulation

In this section, we first introduce the system model, and then formulate the problem of multipath discovery.

A. *Multi-Source VoD in WMN*

Consider the VoD application in a large community network constructed by wireless mesh networking technology. The network is composed of a number of multi-channel multi-interface wireless mesh routers. Each mesh router can establish wireless connectivity with neighboring mesh routers, so that a static multi-hop wireless network is formed. There are some special mesh routers working as gateways, which are directly connected to the Internet. Previous static channel allocation algorithms, such as [18] [19], can be used to assign each interface of each router with a channel so as to minimize the network interference while maintaining the network connectivity.

For some popular videos, the VoD performance can be improved by $P2P$ technology. Whenever a user has requested a video, it registers to the server, so that the server keeps a list of the users that have buffered the video. When a new user visits, he or she queries the server for the list of users, from which they can stream the video. If there are such users, the new user can stream the video not only from the media server, but also from the other peers. For the peers that are located in the community network, the user downloads over a multi-hop wireless path, while for the peers in the Internet, the user downloads over a path, which consists of a multi-hop wireless path from the user to one of the gateways and a path from the gateway to the peer in the Internet.

We call the peer or the server that provides the download of the video as the *sending node*, and the user that requests the video as the *receiving node*. A mesh router is called a ***sender*** (or ***receiver***) if it is connected with one or more sending nodes (or the receiving node). If the sending node is located in the community network, the mesh router that the user is directly connected with is the sender. If the sending node is in the Internet, the gateway through which users in the community network can access it is the sender. In the example illustrated in Fig.1, where there is a media server in the Internet, we assume $P1$, $P2$, A, C are the peers that have buffered the video. If X want to watch the video, there are five available senders in this case, which include $R1$, $R2$, and the three gateways.

In this paper, we focus on multipath routing and rate allocation to improve the video streaming performance in multi-channel multi-radio wireless mesh networks. For each arriving VoD request, we determine the routes from senders to the receiver and the rate on each route for video streaming. A VoD request can be satisfied if we find the routes with the required bandwidth that convey the demanded video quality; otherwise, the VoD request cannot be satisfied, and will be blocked. If a VoD request can be satisfied, the receiver will establish connections for video streaming in the network. For a VoD request, from the time when the connections are established to the time when they are closed due to the end of video streaming, we call the set of connections as a ***VoD session***.

B. Multipath Discovery

Consider a wireless mesh network $G(V, E)$, where V denotes the mesh routers whose locations are known. There is an undirected edge $(u, v) \in E$ if and only if $d(u, v) \leq R$, where $d(u, v)$ is the Euclidean distance between mesh routers u and v, and R is the maximum wireless radio transmission range.

Suppose each mesh router is equipped with Q interfaces, and there are K orthogonal channels. Given a channel assignment A, where $A(u)$ ($u \in V$) denotes the set of channels assigned to the interfaces of u, the *topology* $G_A(V, E_A)$ can then be determined. There is a wireless link $e = (u, v, c) \in E_A$ if and only if $(u, v) \in E$ and $c \in A(u) \bigcap A(v)$, that is, u and v each have at least one interface working on channel c.

We use the protocol model to determine whether two wireless links in G_A interfere with each other or not. For any two links $e_i, e_j \in E_A$, define their distance $d(e_i, e_j)$ as the minimum Euclidean distance between any node of one link and any node of the other link. e_i and e_j interfere with each other if: 1) $c(e_i) = c(e_j)$, where $c(e)$ denotes the channel of wireless link e. 2) $d(e_i, e_j) \leq I$, where I is the wireless interference range, which is usually $2R$.

Definition 1: Given topology $G_A(V, E_A)$, the ***conflicting table*** is the set of all link pairs (e_i, e_j), where $e_i, e_j \in E_A$ and $e_i \neq e_j$, such that e_i and e_j interfere with each other.

In the topology $G_A(V, E_A)$, let $r \in V$ be the receiver, and assume there are n senders $S = \{s_1, s_2, ..., s_n\} \subseteq V$. We consider the case where a mesh router will not be a sender and a receiver at the same time (explained in Section II-A), that is, $r \notin S$.

Definition 2: Given topology G_A and its conflicting table T_c, two paths p and q interfere with each other if there exists $(e_i, e_j) \in T_c$ such that $e_i \in p$ and $e_j \in q$. The ***Maximum Interference Disjoint Paths problem (MIDP)*** seeks the maximum number of edge disjoint paths from S to r, denoted by P, where each path is between a different sender in S and r, such that there does not exist (p_i, p_j), where $p_i, p_j \in P$ and p_i p_j interfere with each other.

In other words, the problem finds the maximum number of paths from the set of senders to the receiver, such that any two paths are independent from each other with regard to wireless interference. Let $n =| S |$ be the number of senders and let $m =| P |$ be the number of paths, we have $m \leq n$. The $MIDP$ problem is NP-hard. It can be proved by reducing the maximum independent set problem, which is NP-Complete, to the formulated problem.

In the real-world VoD applications, we not only want to find more interference disjoint paths from S to r, but also need to guarantee the quality of each path with regard to packet loss, delay, and throughput. There have been many studies on metrics for finding good routes between a single source and a single destination in wireless networks. The $WCETT$ metric [20] is widely used in multi-channel multi-interface wireless mesh networks. It not only considers packet loss and delay, but also accounts for channel diversity in each path so as to reduce intra-flow interference. Therefore, we use this metric to evaluate the quality of each selected path. More specifically, we want to find the maximum number of independent paths, while keeping the $WCETT$ value of each path below a certain threshold. In this paper, we use the algorithm proposed in [21] to find an optimal path based on the $WCETT$ metric.

Another problem in real applications is that there might not be enough completely interference disjoint paths in some cases, because of the limited number of channels and number of interfaces. To take advantage of multipath streaming, we can relax the constraint on the path independency a little bit to allow more paths to be found, which improves the robustness of applications.

Definition 3: Given a set of edge disjoint paths P in topology G_A, consider any link $e \in p_0$ ($p_0 \in P$). The set of paths that interfere with e is $I(e) = \{p \mid p \in (P - p_0) \bigwedge p$ has a link that interfere with $e\}$. The ***path interference*** of link e is defined as $PI(e) =| I(e) |$, which reflects how many other paths in P are interfering with this link. The ***maximum path interference*** of P is defined as $MPI(P) = Max_{p \in P} Max_{e \in p} PI(e)$.

Now we can formally describe the problem by considering both the constraint and relaxation.

Definition 4: Given topology G_A and its conflicting table T_c, the ***COnstrained Maximum INdependent Paths problem (COMINP)*** seeks the maximum number of edge disjoint paths from S to r, denoted by P, such that 1) $MPI(P) \leq \alpha$, where α is the threshold to control the level of independency between paths in P; 2) $WCETT(p) \leq \beta$ for $p \in P$, where β is the threshold to control the quality of each path.

Note that by setting $\alpha = 0$ in the first constraint and

$\beta = +\infty$ in the second constraint, the $COMINP$ problem becomes exactly $MIDP$ problem. Therefore, $COMINP$ is also NP hard.

III. Multipath Discovery Algorithm

In this section, we propose two heuristic algorithms, the Iterative Path Discovery algorithm (IPD) and the Parallel Path Discovery algorithm (PPD). Both algorithms run in a centralized fashion on the receiver.

In wireless mesh networks, the mesh routers usually have minimal mobility. For example, in community networks, the routers are usually fixed on roofs of houses. In addition, due to the overhead of dynamic channel switching, static channel allocation strategies are widely used, in which the channel assignment does not change often. This makes it possible for each mesh router to collect the global knowledge of the network, including each other router's position and channel assignment. This can be done by letting each router broadcast the information to the whole network each time the network topology has been reconstructed or channels have been reassigned, which occurs very rarely. Therefore, each mesh router knows the global topology of the wireless mesh network (wireless links and channels), which makes it possible to use a centralized algorithm on the receiver to find multiple paths from senders to the receiver.

A. Iterative Path Discovery

The Iterative Path Discovery algorithm (IPD) finds paths one by one from the senders to the receiver. In each iteration, we find one path from a sender to the receiver, and then update the topology accordingly. This process continues until no new paths can be found from the remaining topology. There are two critical steps in the algorithm, which we will explain below.

1) Path Selection: Let S be the set of senders and T be the initial topology. Let S' be the set of remaining senders, for which we have not found paths yet, and T' be the remaining topology. Initially, $S' = S$ and $T' = T$. In this step, for each $s \in S'$, we first find a minimum $WCETT$ path p from s to r in T' if such a path exists and $WCETT(p) \leq \gamma \cdot w_T(s, r)$, where $w_T(s, r)$ is the $WCETT$ value of the optimal path from s to r in T and γ is a constant to control the quality of each path (we set $\gamma = 1.5$ in our experiment). Denote the resulting set of paths as P, we then need to decide which path to select from P. As the algorithm aims at discovering as many paths as possible in the final solution, we select a path from P based on the following metric.

We define the *total interference of p in $T'(V, E')$*, denoted by $IF_{T'}(p)$, as the number of edges in T' that interfere with any edge in p. More formally,

$$IF_{T'}(p) = \mid \{e' \mid e' \in E' \bigwedge \exists e \in p : Interfere(e, e')\} \mid$$

Note that $Interfere(e, e')$ is true if $e = e'$.

Therefore, we select the path from P, which has the minimum total interference in T'. The reason is that the selected path will render minimum change in the remaining topology, and thus leave us more flexibility in finding more paths afterwards.

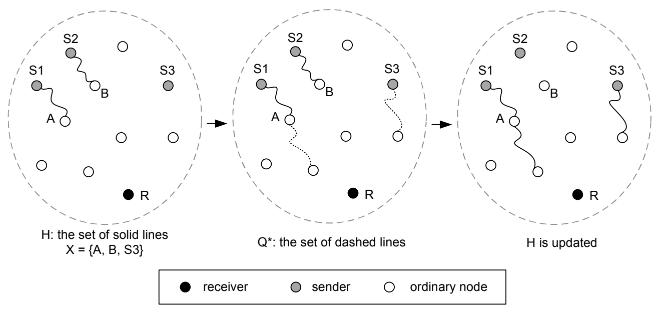

Fig. 2: The Basic Idea of Parallel Path Discovery

2) Topology Update: Once a path p has been selected from T', we need to update T' accordingly. We can guarantee the edge-disjointness of paths by taking off p from T', so that the paths found later will not overlap with the paths previously found. In addition, we need to consider p's interference on T', and guarantee the level of independency among the final set of selected paths.

We use a label l on the edges of T' to record the interference from the already selected paths, where $l(e)$ counts how many selected paths are interfering with link e. At the start of the algorithm, $l(e)$ is initialized to 0. Assume p is the selected path from $T'(V, E')$ in the current iteration. We define the *path interfering set of p in T'*, denoted by $IE_{T'}(p)$, as the set of edges in $(T' - p)$ that interfere with any edge in p.

$$IE_{T'}(p) = \{e' \mid e' \in (E' - p) \bigwedge \exists e \in p : Interfere(e, e')\}$$

We update $l(e)$ for each edge $e \in IE_{T'}(p)$ by increasing 1, indicating that there is one more selected path p that interferes with e. If $l(e) > \alpha$, then we need to take off e from T'. This is because if e has been used in one more path in the remaining topology, then e will interfere with more than α already selected paths, which violates the constraint in the $COMINP$ problem.

The algorithm is simple and takes polynomial running time $O(\mid S \mid^2 \mid E \mid\mid V \mid)$.

B. Parallel Path Discovery

IPD finds a complete path in each step, which limits the search space, and thus may not be able to approximate the optimal solution. Instead of searching for paths one by one, we take a different approach in this section, that is, we perform a parallel search for paths from all senders to the receiver level by level. The basic idea of the Parallel Path Discovery algorithm (PPD) is as follows. In each iteration, we perform a breadth first search from r in the remaining topology $T'(V, E')$, and categorize all the nodes in V into layers based on their distances from r. We start from the senders that are in the farthest layer, and find partial paths that connect as many of them as possible to nodes in the lower layer. We then use these nodes that are reached in the lower

Algorithm 1 Parallel Path Discovery Algorithm

```
1: $T' = T$, $H = \{\}$
2: repeat
3:     Do a breadth first search on $T'$
4:     $Q^* = LayerPathSearch(H, T')$
5:     Extend $H$ and update $T'$ based on $Q^*$
6: until $H$ reaches the receiver $r$
7: Output all the paths in $H$.
```

layer as new senders to take place of the original senders. The process continues until we reach the receiver.

1) Parallel Search: As illustrated in Fig.2, let $H = \{p_1, p_2, ..., p_l\}$ be the set of partial paths found in the current iteration. Each partial path $p_i \in H$ starts from a sender in S, denoted by $s(p_i)$ and ends at a non-receiver node, denoted by $e(p_i)$. We do a breadth first search on the current remaining topology T'. Let $d(v)(v \in V)$ be the distance from v to the receiver r in T'. Consider the set of nodes $X = \{e(p_i)\} + (S - \{s(p_i)\})$, which contains the set of ending nodes of the partial paths and the set of senders that do not have partial paths yet. These are the nodes from which we can further extend the set of partial paths.

In each iteration, we extend partial paths only from the highest layer nodes in X, that is, $X' = \{v \mid d(v) = d_{max}, v \in X\}$ where $d_{max} = Max_{v \in X} d(v)$. We want to find as many paths as possible that connect the nodes from X' to the next lower layer nodes $Y = \{v \mid d(v) = d_{max} - 1\}$. Assume Q^* is the set of paths found to extend H. Let $H_{Q^*} \subseteq H$ be the set of partial paths that can be extended by paths in Q^*. We extend these partial paths and make them as the new H. In addition, we also need to update T'. We take off paths in Q^* together with their interfering edges whose l values exceed the threshold from T', and also recovering paths in $(H - H_{Q^*})$ back into T'. After we have updated H and T', we do a breadth first search again on the updated topology and continue to extend H following the same procedure.

The algorithm is described in Algorithm 1. There is a key sub-algorithm *LayerPathSearch* used to extend the set of partial paths to the next lower layer, which we will explain in the following.

2) Layer Path Search: We first enumerate all the paths of at most 3 hops long from X' to Y in T'. The reason for doing this is that: 1) If we only use one hop path to extend the partial paths, we are always finding the shortest paths. However, in multi-channel wireless mesh networks, the channel diversity is also important for path quality in addition to hop distance. Therefore, we also check 2 or 3 hop paths in order to utilize the channel diversity in the path selection. 2) If we do not limit the path length, we will have exponential search space. In practice, we find that by keeping the threshold at 3, we have enough flexibility in finding channel diverse paths.

Denote the set of paths by Q; we want to find as many paths as possible from Q to extend H without breaking the edge-disjointness and path independency constraints. We rate each

Algorithm 2 FindPathSet(H, T')

```
1: $Q^* = \{\}$
2: In $T'$, find each path $q$ that is at most 3 hops from $X'$
   to $Y$ such that $WCETT(p + q) \leq \gamma \cdot w_T(sender(p), r)$,
   where $p \in H$ concatenates with $q$. Denote the set by $Q$.
3: Rate each path $q \in Q$ by metric $z(q)$.
4: repeat
5:     Take the path $q^*$ with the lowest $z$ value in $Q$.
6:     $Q^* = Q^* + q^*$
7:     Update $T'$ and $Q$ based on $q^*$.
8: until no paths can be selected
9: $Q^*$ is the set of paths to extend $H$
```

path $q \in Q$ based on two factors. 1) The interference caused by q on T', denoted by $IF_{T'}(q)$. The path that causes less interference is preferred to be selected, because it increases the possibility of finding other more paths later. 2) The interference caused by q on the connecting partial path $p \in H$, that is, q starts from the ending node of p. We denote it by $IF_p(q)$. This is related with the channel diversity of each final path discovered. While we want to find more paths, we also need to guarantee the quality of each path. Therefore, q can be evaluated by $z(q) = k_1 \times IF_{T'}(q) + k_2 \times IF_p(q)$.

The algorithm is described in Algorithm 2. We take the path with the lowest z value from Q each time. When we have selected a path $q \in Q$ to extend a partial path $p \in H$, we need to update T' in the same way as described in the iterative path discovery algorithm. We must not only take off q from T', but also take off edges whose l value exceeds the threshold α in T'. As some edges have been taken off from T', we also need to update Q, because some paths may become unavailable in the updated topology. We continue to select paths from Q until Q becomes empty. As a result, we get the set of paths to extend H.

Given H in the current iteration, it is possible that H contains too many partial paths so that they are occupying too many link resources in the lower layers due to wireless interference (these links have been removed from the original topology because their l value exceeds the threshold). This may dramatically reduce the number of paths finally discovered. To deal with this problem, we can release some partial paths from H first, so that we can have enough link resources in the lower layers, and thus we may be able to extend more partial paths to the next layer. The algorithm is described in Algorithm 3, which can be summarized as trying to find a local minimum point.

Let D be the diameter of T. In Algorithm 2, we consider $O(\mid S \mid (\frac{|V|}{D})^3)$ partial paths, and thus the running time is $O(\mid S \mid (\frac{|V|}{D})^3 \mid E \mid)$. Algorithm 3 calls Algorithm 2 $O(\mid S \mid^2)$ times, and thus takes $O(\mid S \mid^3 (\frac{|V|}{D})^3 \mid E \mid)$. There are $O(\mid D \mid)$ iterations in Algorithm 1. Therefore, the algorithm takes polynomial running time $O(\frac{|S|^3 |V|^3 |E|}{|D|^2})$.

Although the parallel path discovery algorithm takes more computation overhead than the iterative path discovery algo-

Algorithm 3 LayerPathSearch(H, T')

```
1:  $H' = H$
2:  $Q^* = FindPathSet(H', T')$
3:  **repeat**
4:      For each $p_i \in H'$, release $p_i$ and update $T'$,
        $Q_i = FindPathSet(H' - p_i, T')$
5:      Let $Q_k$ be the best solution among $\{Q_i\}$
6:      **if** $Q_k$ is better than $Q^*$ **then**
7:          $Q^* = Q_k$, $H' = H' - p_k$
8:      **end if**
9:  **until** no better solution can be found
10: $Q^*$ is the path set to extend $H$.
11:
```

rithm, it has the promise of finding more paths. In the iterative path discovery algorithm, we find one complete path each time, and modify the remaining topology based on the path. As a result, we have made a big change on the remaining topology, and soon no new paths could be found. In contrast, the parallel path discovery algorithm finds partial paths in each step, and leaves more flexibility in the remaining topology for finding more paths.

C. Discussion

If we want to find strictly edge-disjoint paths, the number of paths that can be found will be limited by the number of interfaces that the receiver has. In order to find more paths to enhance the robustness of the application and provide more flexibility for rate allocation, we can relax this constraint by allowing paths to merge at the last hop towards the receiver. Both the iterative and parallel path discovery algorithms can be easily modified to deal with this case. When taking off a selected path from the remaining topology, we do not remove the edge that is directly connected with the receiver. It will be taken off from the remaining topology only when its l value is over the threshold α. In this way, we have still guaranteed the path independency constraint on the last hop.

IV. Joint Routing and Rate Allocation

As the wireless mesh network is designed to be shared among multiple users, the routing and rate allocation for each VoD request should not only consider the performance of itself, but also take into account the existing VoD sessions and the network's ability of satisfying more subsequent VoD requests. Note that the performance of each existing VoD session is dramatically affected by the traffic load on each link used by the session. If the traffic load on a link is high, the application may experience high queuing delay and jitter due to the congestion on this link. In addition, the increase in the number of congested links may disrupt the network's connectivity, thereby causing the network to be incapable of finding routes with required bandwidth for new VoD requests. Therefore, we take our optimization goal as minimizing the network congestion.

Algorithm 4 Joint Routing and Rate Allocation

```
1:  Let $C$ be the capacity of each link, $A(e)$ be the available
    bandwidth of link $e$. Set $Thresh = C$.
2:  Generate sub-topology $T[Thresh]$, which only includes
    links $e$ where $A(e) \geq C - Thresh$.
3:  Find paths $P$ for the session on $T[Thresh]$. Calculate
    optimal rate allocation $r$ on $P$.
4:  Set $P^* = P$, $r^* = r$. Let $v^*$ be the value of the objective
    function under ($P$, $r$) (Equation (4)).
5:  **repeat**
6:      $Thresh = Thresh/2$.
7:      Find paths $P$ for the session on $T[Thresh]$. Calculate
        optimal rate allocation $r$ on $P$.
8:      Let $v$ be the value of the objective function.
9:      **if** $v \geq v^*$ **then**
10:         $P^* = P$, $r^* = r$, $v^* = v$
11:     **end if**
12: **until** (no valid $P$ or $r$ exists || $v < v^*$)
13: return ($P^*$, $r^*$)
```

In this section, we first propose an optimal rate allocation algorithm on the multiple discovered paths, which determines the rate on each path. It is possible that some discovered paths may be unused (or allocated with zero rate), because they are more congested than others. We then propose a joint routing and rate allocation algorithm with the goal of minimizing the network congestion. Unlike previous work [15] [16] [17] that optimize for the performance of a single session only, our optimization not only improves the performance of existing sessions, but also improves the network's capability of satisfying more subsequent VoD requests. The algorithm runs in a centralized fashion on the receiver. We assume each router periodically broadcasts the available bandwidth of its neighboring wireless links to all the routers in the network, so that each router knows the available bandwidth on all the links in the network (similar to [18]).

Assume the multiple paths discovered for a VoD request is $P = \{p_1, p_2, ..., p_m\}$, the receiver needs to determine the optimal data rate on each path. Let r_k be the data rate on path p_k ($k = 1, 2, ..., m$), and R be the total data rate required by the VoD session. We have

$$\sum_{k=1,...,m} r_k = R \tag{1}$$

The traffic load on each link $e_i \in P$, denoted by $t(e_i)$, is

$$t(e_i) = \sum_{p_k \in P \bigwedge e_i \in p_k} r_k \tag{2}$$

Let $A(e_i)$ be the available bandwidth on link e_i. Let IE_{ij} denote whether the two links e_i and e_j interfere with each other. $IE_{ij} = 1$ if it is true, and $IE_{ij} = 0$ if otherwise. The residue capacity of each link $e_i \in P$ under the rate allocation

TABLE I: Video Traces Used in Simulation

Movie Name	Encoding	Quantization Scale	Average Bit Rate (bps)
Silence of the lambs	H.264, Single-Layer	(22, 22, 24)	358365.5
Star Wars IV	H.264, Single-Layer	(22, 22, 24)	373880.4
Die Hard	H.264, Single-Layer	(22, 22, 24)	369614.9

$\{r_1, r_2, ..., r_m\}$, denoted by $Z(e_i)$, is

$$Z(e_i) = A(e_i) - t(e_i) - \sum_{e_j \in P \bigwedge e_j \neq e_i} t(e_j) \times IE_{ij} \quad (3)$$

Therefore, our optimization goal is to maximize the residue capacity of the bottleneck link (or minimize the network congestion).

$$Maximize\ Min_{e_i \in P}\{Z(e_i)\} \quad (4)$$

The problem (1)-(4) is a Max-Min Linear Programming problem, and can be transformed to a general Linear Programming (LP) problem. Therefore, we can solve for the optimal rate allocation $\{r_1, r_2, ..., r_m\}$ in polynomial time. Note that it is not mandatory for all the paths in P to be used for the VoD request. It is possible for some paths to be allocated with rate of zero by solving Equation 4, because they are more congested than others. In other words, *the multipath discovery algorithm gives candidate paths, while the rate allocation algorithm (by solving Equation 4) determines the subset of paths to be used together with the rate on each path for the* VoD *request.*

To approximate the optimal joint routing and rate allocation, we use binary search in Algorithm 4. We use IPD or PPD for path discovery sub-algorithms. They not only find multiple high-quality and independent paths, but also are better at balancing the traffic in the network. By finding edge-disjoint paths, the traffic can be well distributed over the network spatially. By minimizing the interference among paths, the traffic can be well distributed over different channels. In Algorithm 4, we first apply path discovery (IPD or PPD) and rate allocation (LP) algorithms sequentially on the original topology to find an initial solution. We then truncate the topology based on a threshold in each step, that is, we only keep the links with enough available bandwidth. We apply the same sub-algorithms and find new solutions. The search terminates if we cannot find any better solutions by decreasing the threshold.

V. Simulation Methodology

We perform the simulations in $NS2$. The Hyacinth extension [22] has been used to support multiple channels and multiple interfaces per node in the simulator. In all the simulations, we set the maximum radio transmission range to 250 meters and the interference range to 500 meters. We use 802.11 with bit rate of $11Mbps$ for each interface. Unless specifically stated, the simulation is performed in a 60 nodes random topology within an area of $1500m \times 1500m$. Each mesh router is equipped with 4 interfaces, and there are 8 orthogonal channels used in the channel assignment. We use the channel allocation algorithms proposed in [19] to statically assign channels to each interface in order to minimize the wireless interference within the network.

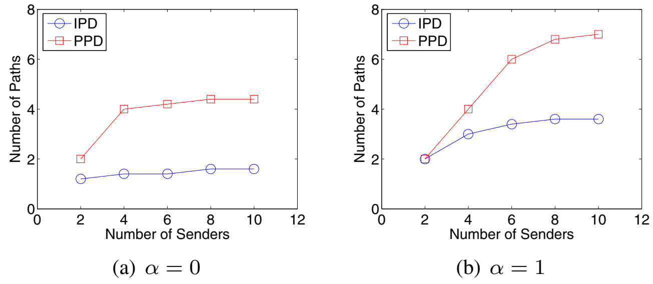

Fig. 3: The Number of Paths Discovered

We use the video traces available in the public domain [23] [24] [25] [26]. Each video trace file includes the size of each encoded video frame together with its quality. During $NS2$ simulation, The frames are encapsulated into UDP packets with the maximum size of 1024 bytes during network transmission and reconstructed at the receiver (similar to [17]). Table I illustrates the three video traces we use for evaluation. Each movie is 30 minutes long. All the traces use GoP (Group of Pictures) length of 16 with I-frame as the first frame in each GoP, and have a rate of 30 frames per second. All the trace files being used contain the frames in the encoder order. In other words, the frames are transmitted through the network in the encoder order.

We compare the following methods of path discovery for multi-source video streaming. 1) $MinW$: find a minimum $WCETT$ path between each sender and the receiver. Thus, if the network is connected and there are n senders, this method will find n paths. 2) $MEDP$: find the maximum number of edge-disjoint paths from the senders to the receiver (Edge disjoint paths have been used in both Internet [27] and multi-hop wireless networks [5]). 3) IPD: use the iterative path discovery algorithm to find the maximum number of independent paths. 4) PPD: use the parallel path discovery algorithm, which has the potential to find more paths than IPD under the same constraints. To be consistent in the comparison, we use the joint routing and rate allocation framework (Algorithm 4) proposed in Section IV for all the methods. In other words, we use Algorithm 4 with different path discovery algorithms as sub-algorithms to find multiple paths, while using the same rate allocation algorithm for all the methods.

VI. Performance Evaluation

In this section, we present the simulation results from $NS2$ based on real video traces.

A. Number of Paths

We select one router in the network as the receiver and randomly designate n routers in the network as senders.

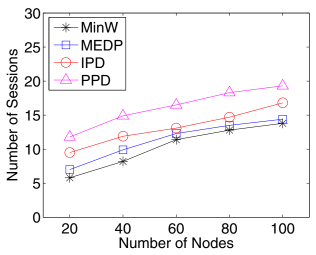

Fig. 4: Maximum Number of Sessions (8 channels)

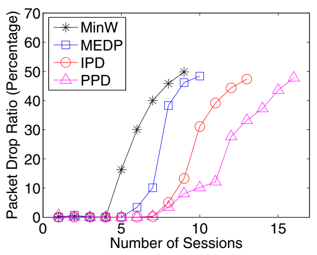

Fig. 5: Packet Drop Ratio (Playout Deadline = 300ms)

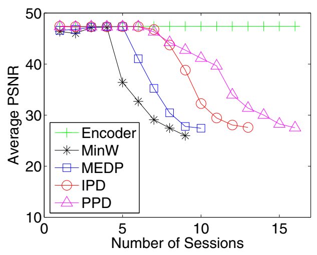

Fig. 6: Average PSNR (Playout Deadline = 300ms)

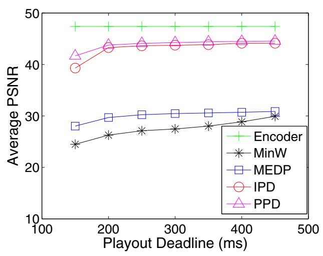

Fig. 7: Average PSNR (8 sessions)

$MinW$ always finds n paths, while the number of paths discovered by $MEDP$ is $min\{n, degree(r)\}$, that is, the network is well-connected so that the number of edge-disjoint paths to the receiver r is bounded by its degree in the topology G_A. For both IPD and PPD, we set the threshold $\alpha = 0, 1$ (see α in Definition 4). By setting $\alpha = 0$, we require that each path does not interfere with any other path. When $\alpha = 1$, each link of any path interferes with at most one other path among the multiple paths finally discovered. According to Section III-C, we allow paths to merge at the last hop towards the receiver for both IPD and PPD. The number of paths discovered by IPD and PPD under different number of senders is illustrated in Fig.3. Each point corresponds to the average of 20 runs. We can observe that PPD is able to discover more paths than IPD under the same constraint. As a result, PPD provides more flexibility for finding appropriate rate allocation to minimize the network congestion than IPD.

B. *Number of Sessions*

We experiment on networks of different scales (with the same density as the default 60 node topology) without changing the other default settings. In each network, three mesh routers are randomly selected as initial senders. They may be mesh routers connected with a peer that has buffered the video, or a gateway through which a local user can access a peer or media server in the Internet. Assume there are 3 popular movies recently (listed in Table I), and each of the initial senders can provide all of the 3 movies. Assume VoD requests arrive in accordance with Poisson distribution. For each arriving VoD request, the user is connected to a random mesh router in the network, and initiates a VoD request for a movie that is randomly selected from the 3 movies. We use different algorithms to find the routing and rate allocation for each arriving VoD request. If the VoD request can be satisfied and has been established successfully, it becomes a sender, which can provide the movie to subsequent VoD requests. This process continues until there is a VoD request that cannot be satisfied. In this way, we can get the maximum number of concurrent sessions that can be supported in the network.

Fig.4 demonstrates the maximum number of concurrent sessions supported by each method under different network scales. For IPD and PPD, α is set to 1. From the figure, we can observe that $MinW$ performs the worst in supporting multiple sessions. $MEDP$ is slightly better than $MinW$. IPD supports over 10 percent more sessions than $MEDP$ on average, while PPD improves the capacity by over 25 percent compared with $MEDP$. This is because IPD and PPD not only find edge-disjoint paths, but also minimize the interference among the paths. Therefore, the traffic of each session can be well balanced over the network both spatially and over different channels. PPD excels IPD, because PPD finds more paths than IPD under the same constraint, and thus PPD provides more flexibility for rate allocation to minimize network congestion than IPD.

C. *Packet Drop Ratio*

When a receiver streams a video over multiple paths, it needs to determine a playout deadline, that is, the time the receiver waits for before playing out the video. As a result, the deadline of each packet to be received can be determined. A packet drop occurs when 1) the packet is lost due to the collision in wireless transmission; 2) the packet has been successfully received, but it is received after its deadline.

In this subsection, we set the playout deadline at the receiver to $300ms$ and calculate the packet drop ratio of VoD sessions. If the receiver streams a video from a gateway, we add a random delay conforming to normal distribution $N(100ms, 20ms^2)$ (the parameters are derived from the packet delays extracted from an experiment where we repeatedly send ping packets from a gateway to a video server) on the path to simulate the delay from the peer in the Internet to the gateway.

Fig.5 illustrates the average packet drop ratio over all sessions when there are different numbers of concurrent VoD sessions in the network. We can observe that the drop ratio increases with the increasing number of sessions, because the network is getting more and more congested. When there are few (less than 5) concurrent session in the network, all methods enjoy very low packet drop ratio. However, when more sessions have been established, IPD and PPD have dramatically lower packet drop ratio than $MinW$ and $MEDP$, because IPD and PPD find better paths for streaming the video. For example, to guarantee an average packet drop ratio less than 20 percent, the maximum number of concurrent sessions that can be supported by $MinW$, $MEDP$, IPD, PPD is 5, 7, 9, 11, respectively.

D. Perceived Video Quality

The PSNR metric of videos perceived at receivers is calculated and demonstrated in Fig.6. In the figure, each point corresponds to the average of PSNR over all the sessions. The '+' line plots the PSNR when there is no packet drop in video streaming (optimal value). The PSNR of the 3 movies in Table I without any frame loss is 47.7 dB, 46.9 dB, and 47.6 dB respectively. We can observe that when there are fewer sessions in the network (less than 5), all the algorithms lead to very high PSNR because of low packet drop ratio. However, when more sessions are established in the network, IPD and PPD obviously excel the other methods, while PPD performs better than IPD. For example, to guarantee an average PSNR no less than 40 dB, the maximum number of concurrent sessions that can be supported by $MinW$, $MEDP$, IPD, PPD is 4, 6, 8, 11, respectively.

Fig.7 illustrates the PSNR under different values of playout deadline when there are 8 concurrent VoD sessions in the network. We can observe that the perceived video quality increases with longer playout deadline. For IPD and PPD, when the playout deadline is over 200ms, the improvement of PSNR becomes less obvious. In comparison, with the increase of playout deadline, there is steady improvement of PSNR for $MinW$ and $MEDP$. This is because IPD and PPD have lower delay jitter than $MinW$ and $MEDP$ in network transmission. Therefore, IPD and PPD only need a small playout deadline to reach near-optimal performance.

VII. Conclusion

In this paper, we first proposed two heuristic multipath discovery algorithms, IPD and PPD, to find multiple independent paths from senders to the receiver for each VoD request. The proposed algorithms consider wireless interference in the multipath discovery, so it is able to balance the video streaming traffic both spatially and on different channels in the network. Based on the multipath discovery algorithms, we then proposed a joint routing and rate allocation algorithm to find the routes and rate allocation with the goal of minimizing the network congestion. We performed simulations in $NS2$ using real video traces, and evaluated the performance of our algorithms using both network layer metrics and application layer metrics for video streaming. Simulation results have shown that IPD and PPD not only increase the maximum number of concurrent VoD sessions that can be supported in the network, but also improve the video streaming quality of each session, compared to previous work. Moreover, PPD achieves better video streaming performance than IPD, because it is able to discover more paths than IPD under the same constraint.

VIII. Acknowledgement

This work was supported in part by the US National Science Foundation under Grants CCF-0514078, CNS-0551464, and CNS-0721441.

References

[1] D. A. Tran, K. A. Hua, and T. T. Do, "A peer-to-peer architecture for media streaming," in *Journal on Selected Areas in Communications, Volume 22*, 2003.

[2] T. Nguyen and A. Zakhor, "Multiple sender distributed video streaming," in *IEEE Transactions on Multimedia, Volume 6, Issue 2*, 2004.

[3] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," in *Computer Networks, Volume 47, No.4*, 2005.

[4] M. R. Pearlman, Z. J. Haas, P.Sholander, and S. S. Tabrizi, "On the impact of alternate path routing for load balancing in mobile ad hoc networks," in *MobiHoc*, 2000.

[5] S. J. Lee and M. Gerla, "Split multipath routing with maximally disjoint paths in ad hoc networks," in *ICC*, 2001.

[6] M. Marina and S. Das, "On-demand multipath distance vector routing in ad hoc networks," in *ICNP*, 2001.

[7] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, 1996.

[8] S. Mao, S. Kompella, Y. Hou, H. Sherali, and S. Midkiff, "Routing for multiple concurrent video sessions in wireless ad hoc networks," in *ICC*, 2005.

[9] S. Mao, Y. T. Hou, X. Cheng, H. D. Sherali, and S. F. Midkiff, "Multipath routing for multiple description video in wireless ad hoc networks," in *InfoCom*, 2005.

[10] W. Wei and A. Zakhor, "Path selection for multi-path streaming in wireless ad hoc networks," in *Proc. ICIP*, 2006.

[11] I. Sheriff and E. Belding-Royer, "Multipath selection in multi-radio mesh networks," in *BROADNETS*, 2006.

[12] D. M. Shila and T. Anjali, "Load aware traffic engineering for mesh networks," in *Computer Communications, Vol.31, Iss.7*, 2008.

[13] D. Li, Q. Zhang, C.-N. Chuah, and S. J. B. Yoo, "Multi-source multi-path video streaming over wireless mesh networks," in *Proc. ISCAS*, 2006.

[14] S. Kompella, S. Mao, Y. T. Hou., and H. D. Sherali, "Cross-layer optimized multipath routing for video communications in wireless networks," in *Journal on Selected Areas in Communications, Volume 25*, 2007.

[15] T. P. Nguyen and A. Zakhor, "Distributed video streaming over internet," in *Proc. SPIE, Multimedia Computing and Networking*, 2002.

[16] T. Nguyen and A. Zakhor, "Distributed video streaming with forward error correction," in *Packet Video Workshop*, 2002.

[17] X. Zhu, S. Han, and B. Girod, "Congestion-aware rate allocation for multipath video streaming over ad hoc wireless networks," in *Proc. ICIP*, 2004.

[18] J. Tang, G. Xue, and W. Zhang, "Interference-aware topology control and qos routing in multi-channel wireless mesh networks," in *MobiHoc*, 2005.

[19] A. P. Subramaniam, H. Gupta, and S. R. Das, "Minimum-interference channel assignment in multi-radio wireless mesh networks," in *SECON*, 2007.

[20] R. Draves, J. Padhye and B. Zill, "Routing in multi-radio multi-hop wirelss mesh networks," in *MobiCom*, 2004.

[21] Y. Yang, J. Wang, and R. Kravets, "Interference-aware loop-free routing for mesh networks," in *UIUC Tech Report*, 2006.

[22] "http://www.ecsl.cs.sunysb.edu/multichannel/."

[23] "http://trace.eas.asu.edu/."

[24] P. Seeling, M. Reisslein, and B. Kulapala, "Network performance evaluation with frame size and quality traces of single-layer and two-layer video: A tutorial," in *IEEE Communications Surveys and Tutorials, Volume 6, No.3*, 2004.

[25] G. V. der Auwera, P. T. David, and M. Reisslein, "Traffic and quality characterization of single-layer video streams encoded with h.264/mpeg-4 advanced video coding standard and scalable video coding extension," in *IEEE Transactions on Broadcasting, Volume 54, Issue 3*, 2008.

[26] G. V. der Auwera and M. Reisslein, "Implications of smoothing on statistical multiplexing of h.264/avc and svc video streams," in *IEEE Transactions on Broadcasting, Volume 55, Issue 3*, 2009.

[27] A. Begen, Y. Altunbasak, and O. Ergun, "Multi-path selection for multiple description encoded video streaming," in *ICC*, 2003.