Prioritized Evolutionary Optimization in Open Session Management for 3D Tele-immersion

Ahsan Arefin Raoul Rivas Klara Nahrstedt
University of Illinois at Urbana-Champaign, Urbana, IL, USA
{marefin2,trivas,klara}@illinois.edu

ABSTRACT

Different 3D tele-immersive (3DTI) activities pose different requirements for application and network level quality of service (QoS) to ensure a strong quality of experience (QoE) for participants. Some applications put heavy weight on audio quality, some consider higher quality for upper body video streams, and some seek very low end-to-end interactivity delay. In addition, a variation in streaming content may arise due to the participants' change of interests (e.g., view change). Therefore, there is a need for an adaptive multi-stream, multi-site 3DTI session management strategy, which is unobtrusive, and optimizes QoS parameters in the 3DTI content distribution based on the user activity and content variation. To address this next generation session management problem, we revisit the design space of multi-stream and multi-site 3DTI session layer. We propose an evolutionary 3DTI session optimization approach using an Open Session Management (OSM) architecture that uses a *global view* of participants and overlay network conditions to optimize prioritized QoS parameters. Experimental results with PlanetLab traces show that the optimization process is computationally unobtrusive, and the optimized TI sessions meet expectations of the participants up to 50% higher compared to the current solutions in the 3DTI space.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]; C.2.2 [**Computer Communication Networks**]; C.2.3 [**Network Operations**]

General Terms

Design, Algorithm, Optimization, Experimentation

Keywords

3D Tele-immersion, Network Protocols, Distributed Application, Session Management, Content Distribution

1. INTRODUCTION

Over the last few years, with the increased high-speed wired and wireless network availability, video traffic has quickly become the dominant fraction of Internet data traffic. Though a significant portion is coming from video-on-demand applications (such as Netflix and YouTube), traffic from interactive 2D video conferencing systems (such as Skype, Google Hangout, and Cisco telepresence) is also very prominent. Moreover, current research efforts from both academia [5, 6, 20, 28] and industry [8, 15, 26] are adopting multi-camera, multi-site, and multi-modal 3D tele-immersive (3DTI) platforms for various online applications such as video conferencing, multi-player gaming, remote learning, collaborative dancing and tele-health.

Even with a large variation in applications, most of the 3DTI systems available on the market are optimized for one particular activity to achieve an optimal performance for the intended activity (e.g., Xbox with Kinect is optimized for gaming). However, recently, we have seen that the same 3DTI platform can be used for multiple activities. An example is shown in Figure 1, where geographically-distributed participants are engaged in video conversation and collaborative dancing using the same 3DTI setup [28].

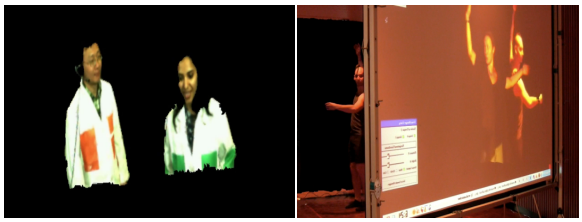

Figure 1: Example of two 3DTI activities: (a) video conferencing, and (b) collaborative dancing.

Different activities require different *minimum quality bounds on quality of service (QoS) values* in network and application levels to meet user expectations [3], which manifest themselves in the form of quality of experience (QoE) [27]. For example, a TI conversation activity requires a high quality audio stream, whereas the minimum quality requirement of the audio stream is low for a virtual lightsaber gaming [2] (where participants virtually fight with each other using lightsabers). However, the lightsaber gaming requires a high quality video stream, and a low end-to-end delay to allow smooth and fast interactions among the participants. Therefore, an adaptive session management is required in 3D tele-immersions that can allocate and re-allocate the QoS parameters at run-time based on the underlying requirements.

Also the priority (i.e., importance) of the QoS parameters varies across the activities. For example, the TI conversation activity puts heavy weight on the audio quality compared to the quality of the video streams. Therefore, to achieve a strong QoE, the content distribution architecture first needs to optimize (maximize) the au-

MMSys'13, February 26-March 1, 2013, Oslo, Norway.

dio quality before optimizing (maximizing) the quality of the video streams subject to the resource availability. On the other hand, a lightsaber activity first needs to optimize (minimize) the end-to-end delay before optimizing (maximizing) the quality of the video streams. Though, here we showed prioritized dependencies between only two QoS parameters, in practice, the priority spans across many. Therefore, a 3DTI session adaptation requires a session optimization based on the *QoS priorities*.

Note that a 3DTI system usually forms a *closed* setup, where participants declare the available list of I/O devices (such as cameras, microphones and haptic sensors) at the beginning of a session. Even though a 3DTI setup is equipped with many I/O devices, not all of them are used for all activities. For example, the TI conversation activity does not use lightsaber sensory streams, which are used to detect the lightsaber's position in the virtual space for the virtual lightsaber gaming. Moreover, the selection of video streams by the participants in the 3DTI space varies based on the participant's view orientation [32]. Therefore, a variation in *streaming content* can arise during the run-time of a 3DTI session, which triggers a session adaptation.

In summary, the streaming content, the minimum quality requirements for network and application QoS parameters and their priorities are highly dependent on the participants' interests in the 3D virtual space and their engaged 3DTI activities. Unfortunately, there is a mismatch between the requirements of TI streaming and today's real-time content delivery infrastructures, both at the network and application levels. Existing real-time content delivery infrastructures do not consider the dynamisms present in users' expectations while streaming to make the best utilization of networking resources [31].

Our overarching goal is to *address the 3DTI session optimization problem as a part of the session adaptation process*. The ultimate goal of the optimization process is to construct a 3DTI content distribution graph considering the content demands and QoS specifications (minimum quality bounds and priorities) in the on-going activities. At this point, we envision that a session manager with a *global view* of the participants and the network condition is a suitable choice for the 3DTI session adaptation. The global view provides several benefits: 1) ensures feasibility in detecting changes in on-going TI activities and content requirements by using a global view of the participants, 2) provides flexibility in optimizing fine-grained session policies considering the global network resources, and 3) improves data plane performance of the participating peers by transferring control-plane loads to a *logically* centralized session controller. To this extent, we propose a centralized 3DTI session management framework, the Open Session Management (OSM) for 3D tele-immersion. OSM uses a global view to construct a multi-site and multi-stream *immersive* 3DTI content distribution topology by optimizing prioritized QoS allocation during session adaptations.

We model the problem of *prioritized QoS optimization* in the multi-site multi-stream content routing topology as a priority-based *multi-objective optimization (MOOP)* problem, given multiple constraints such as stream demands, minimum quality bounds of QoS values, QoS priorities and bandwidth limitations. However, the constraint-based multi-objective optimization in a network routing is an NP-complete problem [10, 25] and hence, we need a heuristics-based solution. We choose a *prioritized evolutionary algorithm* as our heuristics-based solution in this work.

Evolutionary algorithms (EAs) are well-known heuristics to solve MOOP problems. They use an iterative approach and therefore, a potential drawback may include a large latency in the search of an optimal solution. However, using a careful selection of initial seeds (i.e., multi-stream multi-site 3DTI content routing topologies satisfying minimum QoS quality and resource constraints) in the search space and allowing topology diversities in the optimality search, we show that the evolutionary optimization of the prioritized QoS allocation converges very fast. Moreover, the evolutionary algorithms generate intermediate results in the process of searching an optimal solution, which allows us to terminate the algorithm at any time in case of a large latency and use the best generated solution.

We implement a prototype of the OSM architecture and show the feasibility of immersive 3DTI session optimizations using bandwidth and end-to-end delay traces from PlanetLab [21] nodes for 3 to 10 participating sites. The limitation in the number of participants comes from two observations: 1) human attention space is intrinsically limited [32], so, we do not expect the size of an interactive session to be larger than 10 sites, and 2) the number of participants that can be displayed in the virtual space is limited due to the pixel space limitation of the physical display [1].

From the experiments, we show that the evolutionary session optimization provides up to 50% improvements in the allocation of desired QoS parameters, compared to the current content distribution solutions in the immersive 3DTI space. Most of the cases, the optimization process takes less than 500ms, though the complete session adaptation (triggered by the change in activity or user interest) including the selection of QoS bounds, their optimization and finally setting up a new session at the participants can take up to 2 seconds depending on the system load (e.g., number of sites and number of streams per site). However, the adaptation latency can be decreased by reducing the quality of the solution (i.e., having less iterations in the evolutionary optimization) or it can be masked using pre-computation of the optimized topology by predicting future changes in user's interest and intended activity.

2. RELATED WORK

The content routing considering QoS optimizations, is not new to the 3DTI applications. ViewCast [32], for example, constructs a multi-stream, multi-site content distribution graph by maximizing the number of video streams per participant in order to improve the 3D video quality. They consider stream priority (defined by the participant's view orientation) while dropping the stream requests given the bandwidth and delay constraints. Wu et al. [28], on the other hand, constructs a randomized content distribution graph. They consider node (3DTI participant) *criticality* while dropping streams, where the criticality for each remote node is computed as the reciprocal of the number of streams requested from that node. When a conflict occurs due to the resource constraints, the streams from the lower critical nodes are dropped. They do not guarantee the maximization of the higher priority streams, rather they maximize the average number of streams from unique nodes (participants). Both [32] and [28] consider the QoS maximization in terms of the number of video streams in the content distribution architecture. However, maximizing the number of video streams is not the only QoS factor that controls participants' QoE. Other QoS parameters such as perceptual quality of the streams, stream frame rate, end-to-end delay, number of multi-modal channels and synchronization skew highly influence the 3DTI experience [3]. Sometimes the dropping of a video stream to optimize the rate of another stream can gracefully improve the QoE. In this paper, our goal is to find an optimal 3DTI overlay distribution graph considering multiple dimensions of QoS specifications that impact the participant's QoE.

There are also other works that consider constraint-based QoS maximization in network routing. Celerity [30], for example, constructs a multi-party multi-rate 2D video conferencing solution sub-

ject to bandwidth and end-to-end delay constraints. They formulate the problem as a rate maximization problem and provide a polynomial-time tree packing algorithm on the source and an adaptive rate control along each overlay link. However, they do not prioritize streams that share common resources and therefore, it is not applicable in the 3DTI domain. Some related literature can also be found on QoS routing for IP multicast [29] and overlay construction for application-level multicast (e.g., [19, 16, 23]). However, none of them consider the complexity in constructing multi-stream dissemination structures, where the participating sites and resources are shared. Moreover, the priority of the QoS parameters in the 3DTI space varies depending on the activity a participant performs in the virtual space.

The use of evolutionary algorithms (EAs) is not new in real-time applications. Kumar et al. [10] uses an evolutionary algorithm to construct a network path considering priority among data rate, path reliability, network bandwidth and end-to-end delay for different applications. Huang et al. [17] uses an evolutionary algorithm to search an optimal playout buffering size and Xiang et al. [13] proposes an evolutionary QoS routing for ATM networks.

3. SYSTEM MODEL

We present 3DTI system model in this section. A list of symbols and denotations is shown in Table 1.

Table 1: Notations and definitions

Notations	Definitions
s_i	i^{th} input stream
S	Set of all streams in a 3DTI session
x_i	i^{th} cQoS parameter
r_i	Request stream set of Site-i
o_i	Generated stream set from Site-i
$G(V, E)$	Network graph with participant set V and network links E
v_i	Gateway of Site-i indicating a vertex in V
ibw_v	Inbound bandwidth capacity of v
IBW_v	Allocated inbound bandwidth to v
obw_v	Outbound bandwidth capacity of v
OBW_v	Allocated outbound bandwidth to v
e_{ij}	Connecting network path between v_i and v_j
d_{ij}	End-to-end delay associated to e_{ij}
α	3DTI activity
min_x^α	Minimum quality of x for activity α
max_x^α	Maximum quality of x for activity α
p_x^α	Priority of x for activity α
c_i	i^{th} chromosome (or distribution graph)
ρ_i	Rank vector of chromosome c_i
R_s	Allocated rate for stream s
NVS	Number of video streams received per remote site
NSC	Number of stream channels received per remote site

3.1 3DTI System Model

3D tele-immersive system is a distributed platform connecting multiple remote sites containing a large number of input and output devices (as opposed to the traditional video conferencing solutions) into one virtual shared space as shown in Figure 2. The system is usually composed of three architectural tiers: the capturing tier, the data dissemination tier and the rendering tier.

In the **capturing tier**, multiple capturing devices such as 3D cameras (capturing separately the upper body and the lower body of the participants), microphones, heart rate monitors and mobile gyroscope capture the cyber-physical multi-modal information of each participant at his/her physical site. The captured streams are sent to a local *rendezvous point*, called site gateway, which is responsible for data dissemination. The **dissemination tier** consists of overlay network of gateways multiplexing streams to and from each site. In the **rendering tier**, multiple rendering devices such as video displays, heart rate viewer, audio speakers and mobile vibrators are used. Therefore, a 3DTI system truly represents a multi-site, multi-stream and multi-modal environment.

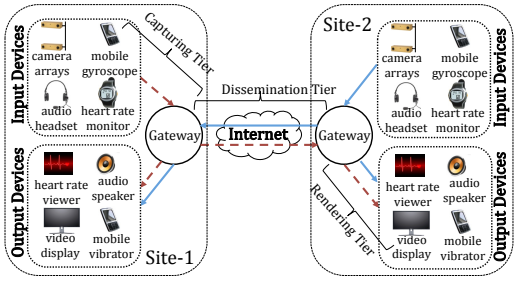

Figure 2: Architecture of a 3DTI system with two sites.

3.2 Data (Stream) Model

Each input device generates a stream s in the capturing tier. Not all input streams of a participating site are required by remote participants. The demand for video streams by a remote participant depends on the view orientation of the participant [32]. For example, if a viewer is currently looking at the front of a 3D object, video streams generated by the back cameras are less important and can be dropped. The demands for other streams by the participants are usually given by the TI activity. For example, a video conversation activity does not require a stream from the heart rate monitor, however a tele-health session between doctors and patients may use it. Therefore, the set of streams requested by each participant can be different based on the engaged activity and interest.

The priority (importance) of a stream inside a request set can be computed by using the view orientation and the camera orientation (for the video streams [32]) as well as using the domain knowledge of the activity (for multi-modal streams). For example, video streams from the frontal cameras are more important than the video streams from the cameras which are 45^0 from the frontal position. On the other hand, audio streams are more important than video streams in a TI conversation and a video stream is more important than an audio stream in a collaborative dancing.

The list of streams requested by a remote participant (e.g., Site-i) can be represented by $r_i = \{s_1, s_2, \cdots, s_k\}$ where the remote participant requests k streams. The set size of r_i can vary dynamically during a session run-time if the participant changes view or engages in a new TI activity. The stream request r_i is different from request r_i', i.e., $r_i \neq r_i'$ if $\exists s_1 \in r_i \wedge \exists s_2 \in r_i'$ such that $s_1 \neq s_2$. Therefore, changing stream requests causes a TI session to discard old streams and generate new streams in the content distribution architecture. Note that we use *color-plus-depth* 3D images [27] constructed from binocular passive stereo as video streams and stereo audio as audio streams.

3.3 QoS Metadata Model

We consider two types of QoS parameters (metadata) in the 3DTI space: *controllable QoS* (cQoS) metadata and *derived QoS* (dQoS) metadata. cQoS are the metadata that an application can control by configuring the content distribution topology and application parameters, for example, end-to-end delay (EED), bit rate[1] of a stream s (R_s), number of video streams to accept from each remote site in order of priority (NVS) and number of total streams to accept from each remote site in order of priority (NSC). If

[1]The stream bit rate can be defined using the media quality (such as the color-plus-depth level-of-detail for 3D video [27] and perceptual evaluation of speech quality [24] for audio) and the media frame rate.

Table 2: cQoS specification of TI activities (rates in kbps and delays in ms)

Property	TI conversation	TI virtual lightsaber
cQoS Bound	$128 \leq R_{s_{audio}} \leq 256$, $R_{s_{light}} = 0$, $2 \leq NSC \leq 5$, $1 \leq NVS \leq 4$, $EED \leq 500$, $1500 \leq R_{s_{ub\_H}} \leq 2500$, $1000 \leq R_{s_{ub\_L}}, R_{s_{lb\_H}}, R_{s_{lower\_body\_L}} \leq 2500$	$64 \leq R_{s_{audio}} \leq 256$, $5 \leq R_{s_{light}} \leq 10$, $4 \leq NSC \leq 6$, $2 \leq NVS \leq 4$, $EED \leq 200$, $1000 \leq R_{s_{ub\_H}}, R_{s_{ub\_L}}, R_{s_{lb\_H}}, R_{s_{lb\_L}} \leq 2500$
cQoS Priority	$R_{s_{audio}} > R_{s_{ub\_H}} > EED > NVS > R_{s_{ub\_L}} > R_{s_{lb\_H}} > R_{s_{lb\_L}} > NSC > R_{s_{light}}$	$EED > R_{s_{light}} > R_{s_{ub\_H}} > R_{s_{lb\_H}} > NVS > R_{s_{ub\_L}} > R_{s_{lb\_L}} > NSC > R_{s_{audio}}$

x_i indicates the i^{th} cQoS metadata, the set can be represented as $X = \{x_1, x_2, \cdots, x_m\}$ for m number of cQoS metadata.

There are some QoS metadata over which the session layer does not have any direct control, for example, CPU utilization of the gateway, 3D rendering time, application frame size, audio silence period, participants view orientation and so on. We call them dQoS metadata. However, they are important for identifying the physical characteristics of the participants.

3.4 Activity Model

Activities in the 3DTI space can be defined using the movement-based characteristics of the participants. Each activity α specifies a minimum quality bound min_x^α for each cQoS parameter $x \in X$ and assigns a priority (importance for activity α) p_x^α to it. For m cQoS parameters, $p_x^\alpha = m$ indicates the highest priority and $p_x^\alpha = 1$ indicates the lowest priority. The maximum quality bound max_x^α of the cQoS parameter x depends on the application design (e.g., video coding) and the device characteristics (e.g., maximum captured frame rate) and they do not vary across activities.

The cQoS specifications (i.e., the quality bounds of the cQoS parameters and cQoS priority) of an activity can be obtained using a systematic subjective and objective evaluation for that activity. Interested readers can be redirected to [3]. Below we present an example of cQoS specifications for two 3DTI activities: 1) TI conversation, and 2) TI virtual lightsaber fight.

In a TI conversation activity, participants are engaged in a video conversation, where they mainly focus on the audio (s_{audio}) and the highest priority upper body video (s_{ub_H}) of the remote participants (the priority of the video streams are measured using the view orientation). Therefore, to ensure a strong QoE in the conversation session, the application first needs to ensure the high quality (i.e., high bit rate) delivery of s_{audio} and s_{ub_H} before ensuring the delivery of other streams given the resource limitation. The maximum end-to-end delay (EED) for all streams can be relaxed compared to the other collaborative activities, however it needs to be optimized before considering the optimization of the quality of the lower priority upper body video streams and any lower body video streams. Another important cQoS parameter is NVS. We want to maximize the number of video streams (in order of stream priority) each site receives (at the minimum rate) before maximizing the rate of the low priority streams. Since, for TI conversation, we do not have any other sensory channels except audio and video, the maximization of NSC is not important.

In a virtual lightsaber fight, each participant uses a lightsaber sensor (to indicate the lightsaber position in the virtual space) to virtually hit other participants and gain points to win. For a successful lightsaber session, we require both the highest priority upper body video stream (s_{ub_H}) and the highest priority lower body video stream (s_{lb_H}) to represent the full human body along with s_{audio} and the lightsaber sensory stream (s_{light}). Also, EED consideration is more critical in the lightsaber game compared to the conversation activity, because, having a EED may impact the hitting efficiency of the participants. Therefore, before maximizing the quality of the video streams, the application first ensures the lowest possible EED. The quality of the lightsaber sensory stream is important for detecting hitting accuracy in the virtual space. Also, increasing the number of video streams (NVS) improves the lightsaber gaming experience. Since, the minimum specification of NSC already considers the inclusion of lightsaber stream, audio stream, and the highest priority video streams, maximization of NSC is less important (note that the maximization of number of video streams is already covered by NVS). The audio quality is considered as the least important cQoS parameter for this activity.

Based on the above definitions, we give an example of cQoS quality bounds and priority orderings. We assume that each site contains 2 upper body video streams (spaced 90^0 from each other), 2 lower body video streams (vertically aligned with the upper body video streams), 1 lightsaber sensory stream and 1 audio stream. The cQoS parameters are: average rate of audio streams ($R_{s_{audio}}$), average rate of highest priority upper body video streams ($R_{s_{ub\_H}}$), average rate of lowest priority upper body video streams ($R_{s_{ub\_L}}$), average EED, average NVS, average rate of highest priority lower body video streams ($R_{s_{lb\_H}}$), average rate of lowest priority lower body video streams ($R_{s_{lb\_H}}$), average rate of light sensory stream ($R_{s_{light}}$), and average NSC. The averages are taken over all stream requests from all participating sites. Table 2 shows the quality bounds as well as the priorities for 9 cQoS parameters for both TI conversation and virtual lightsaber. The maximum bit rates of the streams are computed using the maximum media quality and the maximum media frame rate allowed in our 3DTI setup.

4. FRAMEWORK FOR 3DTI ADAPTIVE SESSION OPTIMIZATION

In this section, we provide an overview of the design space and then propose the OSM architecture for immersive 3DTI session optimization.

4.1 Design Space

Why do we trigger session adaptations? Our goal is not to provide any specific classification of TI activities, rather based on our previous results in [3], we argue that different activities present different expectations of the cQoS specifications to the participants. In addition to the activity dependency, a variation in the 3DTI streaming content can occur due to the view change (or interest change) by the participants in the 3DTI space. Therefore, an adaptive session management is required to reflect the changes (activity change or interest change) made by the participants to ensure a strong QoE.

When do we trigger session adaptations? At this point, we assume that the notification of the activity change is given manually, however the view change of the participants are detected using the stream differentiation technique proposed in [32]. Both of them are called *session updates*. A session adaptation is triggered when the system experiences a session update. The session updates are less frequent (usually in the order of several minutes to hours) in 3DTI applications.

What does happen during session adaptations? When a session update is triggered, the session state(s) are modified; hence, the requirements on the cQoS parameters change. However, maintaining updated minimum quality of cQoS parameters in the new 3DTI

session only guarantees acceptable QoE to the participants. To improve the perceivable QoE with maximum utilization of the resources, an optimization of cQoS values is required. However, different TI activities put different priorities in the cQoS optimization (shown in Table 2). Therefore, we need a prioritized cQoS optimization in the adaptation process. Finally, we can formally define the 3DTI adaptive session optimization as the process of constructing a content distribution graph for multi-site multi-stream 3DTI contents (triggered by a session update) that optimizes the cQoS values in order of their priorities subject to their minimum quality bounds and network resource constraints. The pipeline of the 3DTI adaptive session optimization is show in Figure 3. The constructed overlay structure persists until a new session update arrives.

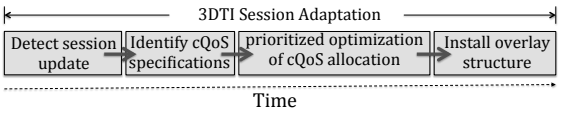

Figure 3: 3DTI session adaptation pipeline.

In this paper, we only consider the optimization step in the 3DTI session adaptation process. The 3DTI session optimization is challenging because of two reasons. First, the optimization process needs to consider optimization of cQoS parameters in order of the activity-defined priorities, and second, the process should maintain the minimum quality constraints defined by the activity while constructing a multi-site and interest-based multi-stream content distribution graph. Both of the challenges are NP-complete. Therefore, we need to design heuristics-based solutions.

4.2 Open Session Management Architecture

A high level architecture of OSM is shown in Figure 4. Each participant gateway maintains a lightweight session layer, called *local session controller* (LSC). A global view of participating peers and networking infrastructures are maintained at a centralized session controller, called *global session controller* (GSC).

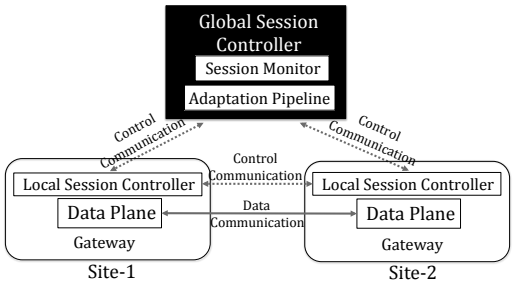

Figure 4: Open Session Management architecture.

Global Session Controller. The GSC is responsible for adaptive session optimization (that uses the pipeline processes shown in Figure 3) at the *Adaptation Pipeline* shown in Figure 4. It constructs a global view of the network resources and the participants' requests using a *Session Monitor* component. After the construction of an optimized topology, the topology information is sent to the LSCs of the participating sites.

Local Session Controller. On the other hand, the LSC is responsible for maintaining the policy (e.g., enforcing the constructed distribution topology) at the participants. It also helps the session monitor component at the GSC with the collection of metadata about participants' current views and local network conditions.

Since our focus in this paper is to design algorithms for the 3DTI session optimization that runs inside the GSC, for the rest of the paper, we concentrate on the session optimization problem.

5. SESSION OPTIMIZATION

In this section, we formalize the optimization problem and then discuss our solutions.

5.1 Problem Formulation

5.1.1 Input

A 3DTI system with N participants can be represented as a complete graph $G = (V, E)$, where $V = \{v_i\}$ is the set of gateways and $E = \{e_{ij}\}$ is the connecting network path from gateway v_i to v_j for $1 \leq i, j \leq N$ and $i \neq j$. Each vertex (site gateway) v is defined by a set of output streams (o_v) it generates, a set of request streams (r_v) it demands, an inbound bandwidth capacity (ibw_v) and an outbound bandwidth capacity (obw_v). Each edge e_{ij} is defined by one way end-to-end delay (d_{ij}) between vertex v_i and vertex v_j.

For ease of understanding, we use s_i^v to represent the i^{th} stream generated by vertex $v \in V$. Also, we assume that the set r_v is ordered in the descending order of stream priority (mentioned in Section 3.2) and S is the set of all streams available in the system, i.e., $S = \{s_j^v\}, \forall v \in V$ and $1 \leq j \leq |o_v|$. For the current activity α, the priorities of the m cQoS parameters are defined as $x_1 > x_2 > x_3 > \cdots > x_m$ (i.e., $p_{x_1}^\alpha > p_{x_2}^\alpha > p_{x_3}^\alpha > \cdots > p_{x_m}^\alpha$).

5.1.2 Optimization Goal

The goal of session optimization is to construct a directional multi-stream and multi-site 3DTI content distribution graph $G' = (V', E')$. Each directional edge is associated with a stream and a rate, and each vertex is defined by an inbound and outbound bandwidth allocation for activity α that optimizes $x \in X$ in order of their priorities, subject to the following constraints:

- **Resource (Bandwidth) Constraints:** If IBW_v and OBW_v are the inbound and outbound bandwidth allocations, respectively, for vertex $v \in V'$ in graph G', then,
  $\forall\, v \in V',\ \ IBW_v \leq ibw_v$ and $OBW_v \leq obw_v$,
- **cQoS Constraints:** $\forall x \in X,\ \ x < | > | = min_x^\alpha$

The problem clearly represents a priority-based multi-objective optimization (MOOP) problem [10]. However, solving MOOP for a network graph even for a single source and destination with a single stream is an NP-complete, in fact it is considered as an intractable problem for large networked systems [14, 25]. Heuristics for solving the MOOP problem have been proposed since the constraint-based routing algorithms was discovered; however they are limited, even concievely missing [7, 12].

5.2 Priority-based Multi-objective Session Optimization

Evolutionary algorithms (EAs) have emerged as powerful heuristic tools for solving NP-complete and NP-hard problems and have received a great attention because of their ability for solving multi-objective optimizations in many domains such as network routing [10, 13] and real-time playout scheduling [17]. In this paper, we adopt genetic algorithm (GA) (a genre of evolutionary algorithms) as a heuristic algorithm to solve priority-based multi-objective optimization problem in 3DTI multi-site and multi-stream content distribution.

Genetic algorithms have derived their inspiration from the process of natural evolution and represent an iterative procedure of applying basic genetic operators over the candidate solutions. These genetic operators include: *selection*, *crossover* and *mutation*. The process starts with a set of initial solutions as the candidate solutions and then applies generic operator on them in an iterative fashion to find an optimal solution. The following subsections describe

these operators and the associated genetic algorithm in the 3DTI space.

5.2.1 Encoding

Instead of binary encoded GA, we use a real parameter GA. A global solution graph (called *chromosome*) for multi-stream multi-site content distribution can be broken down into individual subgraphs representing the distribution of each stream in the system. If $G'_s = (V'_s, E'_s)$ defines the solution subgraph for the distribution of stream s, a chromosome (or global distribution graph) representing the distribution of all streams $s_i \in S$ can be represented by $G'_{s_1}|G'_{s_2}|\cdots|G'_{s_k}$, where k is the number of total streams available in the current 3DTI session. If a stream s is dropped in a 3DTI session, an empty subgraph is represented by G'_s, where $E'_s = \phi$.

An example of chromosome encoding is shown in Figure 5 with 3 sites (vertices are represented as A, B and C); each generating 2 streams (represented as s_1^A, s_2^A, s_1^B, s_2^B, s_1^C and s_2^C, respectively). In practice, the number of streams generated from each site can be much higher. Suppose, the request sets from the participants are, $r_A = \{s_1^B, s_1^C, s_2^B, s_2^C\}$, $r_B = \{s_1^C, s_1^A, s_2^C, s_2^A\}$ and $r_C = \{s_1^A, s_1^B, s_2^A, s_2^B\}$, where streams are listed in-order of their priorities. A sample multi-stream multi-site distribution graph (G') is shown in Figure 5 (a) and the corresponding subgraphs for individual streams are shown in Figure 5(b-g). Therefore, the chromosome representing G' can be encoded as :
chromosome (c) = $G'_{s_1^A}|G'_{s_2^A}|G'_{s_1^B}|G'_{s_2^B}|G'_{s_1^C}|G'_{s_2^C}$.

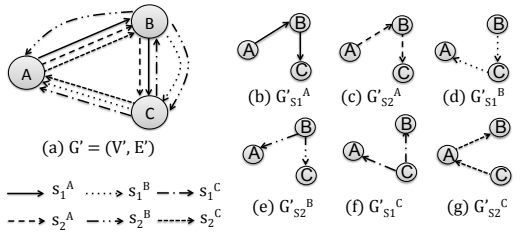

Figure 5: (a) Example of a multi-site multi-stream distribution graph (chromosome), (b)-(f) multi-site distribution subgraphs for individual stream (components of chromosome).

5.2.2 Initialization of Solution

The initialization of solution is very important for GA to ensure faster convergence towards the near optimal solution. To keep the search space and convergence time bounded (required in the interactive 3DTI space), we only consider *valid* chromosomes (i.e., valid content distribution graphs) in each iteration of the algorithm. A valid chromosome is defined as a multi-site, multi-stream distribution graph, where each cQoS metadata maintains at least the *minimum quality* subject to the resource (bandwidth) availability. However, the minimum quality problem for finding multicast trees subject to the bandwidth constraint is also an NP-complete problem [4, 22].

To construct an heuristic-based solution for finding the initial set of chromosomes with minimum quality, we use ViewCast [32]. To understand how ViewCast generates the initial solutions, we first define the minimum quality problem as follows.

Minimum Quality Problem. Construct a multi-stream multi-site 3D content distribution graph for activity α, that

1. satisfy $\forall x \in X,\ x < |>| = min_x^{\alpha}$,
2. subject to $\forall\ v \in V',\ IBW_v \leq ibw_v$ and $OBW_v \leq obw_v$

Minimum Quality Guarantee Using ViewCast. Suppose, we need to forward a stream s to a node v in the process of constructing a global distribution graph as an initial solution. V_s is the set of vertices currently holding stream s due to the current distribution topology. A vertex $v_i \in V_s$ is true if one of the following two conditions is satisfied: 1) v_i is the source of s (i.e., $s \in o_{v_i}$), or 2) v_i receives s via previously assigned network paths in the graph. Therefore, all $v_i \in V_s$ are the candidates for forwarding streams s to v. We randomly select a vertex $v_i \in V_{s_i}$ provided that both v_i and v have available bandwidth to satisfy the minimum bit rate of s and the establishment of the path from v_i to v does not violate the end-to-end delay. Similarly, the distribution graphs are constructed for all streams requested by all vertices.The process of initial solution (the complete content distribution graph) generation is shown in Algorithm 1.

Algorithm 1 Generation of Initial Solution Using ViewCast

```
Input: G = (V, E) and r_{v_i}, ∀v_i ∈ V
Algorithm:
set chromosome c = φ, |r| = max. size of request set;
set V'_{s_i} ← vertex that originates stream s_i;
Randomly shuffle ordering of vertices in V
for (int index ← 0; index < |r|; index ← index + 1) do
  for all v_i ∈ V do
    s_i ← r_{v_i}[index];
    if s_i ≠ null then
      v_j ← get random vertex ∈ V'_{s_i}
      if (e_ji) does not violate (1) & (2) in min quality problem then
        V'_{s_i} ← V'_{s_i} ∪ v_i;  E'_{s_i} ← E'_{s_i} ∪ e_ji;
      end if
    end if
  end for
  c ← c ∪ (V'_{s_i}, E'_{s_i});
end for
return;
```

Example. Let us consider the 3DTI setup discussed in Section 5.2.1. The cQoS parameters we use here are R_s ($\forall s \in S$), EED, NVS and NSC. The process of constructing the distribution overlay in ViewCast is iterated from the highest priority stream to the lowest priority stream for each vertex for a given vertex order. If we consider a vertex order of $\{A, B, C\}$ (in Algorithm 1), the ViewCast constructs a distribution subgraphs for streams in the following order: $s_1^B \rightarrow A$ (s_1^B to A), $s_1^C \rightarrow B$, $s_1^A \rightarrow C$, $s_1^C \rightarrow A$, $s_1^A \rightarrow B$ and so on. If we consider vertex order of $\{B, C, A\}$, then the stream order to construct subgraphs is $s_1^C \rightarrow B$, $s_1^A \rightarrow C$, $s_1^B \rightarrow A$, $s_1^A \rightarrow B$, $s_1^B \rightarrow C$ and so on. Figure 6(a) shows the construction of the distribution graph for stream s_1^B. Suppose, the ViewCast already assigns an edge e_{BA} to satisfy the demand of s_1^B to vertex A. To satisfy the request for vertex C, we can get s_1^B from either vertex A, or vertex B. If both paths e_{BC} and e_{AC} have available bandwidth higher than the minimum value of $R_{s_1^B}$ and they do not violate the minimum bound of EED, any of these two paths can be selected randomly. However, if neither of them satisfies the bandwidth or delay constrains, the stream request is dropped. If the dropping of the stream violates minimum bounds of NVS or NSC, then the solution is rejected. A new solution is started using another random ordering of vertices.

If we assume that both e_{BC} and e_{AC} are valid paths in Figure 6(a), then Figure 6(b) and Figure 6(c) generate two different subgraphs ($G''_{s_1^B}$ and $G'''_{s_1^B}$, respectively) which eventually contribute to two different global multi-stream multi-site content distribution graphs (chromosomes) as shown in Figure 6(d) and Figure 6(e), respectively. Here, we assume that subgraphs for other streams are unchanged. Therefore, by using random ordering of vertices and random selection of stream sources, we can generate different unique solutions (chromosomes). These solutions are then added into a sample pool for selection.

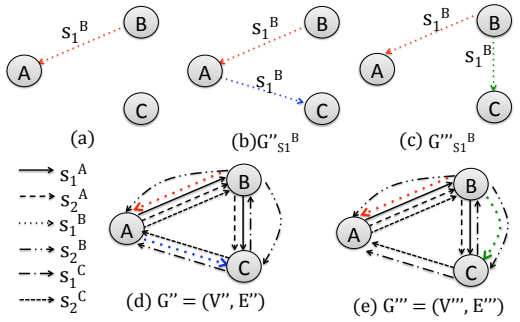

Figure 6: (a) Edge e_{BA} is already assigned, i.e., $V_{s_1^B} = A, B$, (b) subgraph $G''_{s_1^B}$ is generated if vertex C gets s_1^B from vertex A, and (c) subgraph $G'''_{s_1^B}$ is generated if vertex C gets s_1^B from vertex B, (d) distribution graph G'' using subgraph $G''_{s_1^B}$, and (e) distribution graph G''' using subgraph $G'''_{s_1^B}$.

5.2.3 *Selection based on cQoS Priority*

The concept of priority-based MOOP lies in the selection process. A priority-dependent selection scheme, based on *tournament selection* [11], is used for the selection process of chromosomes from the sample pool. As we explained earlier, each chromosome represents a multi-site multi-stream distribution graph satisfying the minimum quality of the activity requirements and bandwidth constraints. Using the subgraphs available in the chromosome, we can construct a global graph and measure the average values of each cQoS parameter (an example is shown in Section 6). Each chromosome is then *ranked* against all other chromosomes based on these average cQoS values in order of cQoS priorities. To allow variation (known as "chromosome diversity") in the selection process, we also consider *crowding distance* [11] of the chromosomes in the comparison. Below we first discuss how to compute the rank ρ_i and the crowding distance γ_i for a chromosome c_i.

Rank. The rank is a vector containing the numerical values of the cQoS parameters and can be represented as $\rho = [|x_1|, \cdots, |x_m|]$, where $|x_i|$ is the value of cQoS parameter x_i. We assume that the cQoS parameters are prioritized as $x_1 > x_2 > x_3 > \cdots > x_m$. If we have two rank values $\rho_i = [|x_1|, \cdots, |x_m|]$ and $\rho_j = [|x'_1|, \cdots, |x'_m|]$ for chromosomes c_i and c_j, respectively, the relationships between the rank values can be represented as follows:

- $\rho_i > \rho_j$ if and only if $|x_t| > |x'_t|$ and $\forall (1 \leq q < t) \quad |x_q| = |x'_q|$.
- $\rho_i = \rho_j$ if and only if $\forall (1 \leq q \leq m), \ |x_q| = |x'_q|$.

Crowding Distance. The crowding distance γ_i of chromosome c_i is a measure of the objective space around c_i, which is not occupied by any other solution in the current solution set. It helps to avoid local optima in the search space. To measure the crowding distance for chromosome c_i, we sort the population set based on their rank values and then use the following equation on the ordered set of chromosomes: $\gamma_i = \sum_{x \in X} (|x|_{c_{i+1}} - |x|_{c_{i-1}})$, where $|x|_{c_{i+1}}$ and $|x|_{c_{i-1}}$ are the values of x for chromosomes c_{i+1} and c_{i-1}, respectively, from the sorted set. The crowding distance for the first and last chromosome in the set is set to infinity (∞).

Selection Process. Once we compute the rank value and the crowding distance, the comparison in the selection process works in three steps: (1) chromosome c_i wins over chromosome c_j (i.e., $c_i > c_j$) if $\rho_i > \rho_j$, (2) if $\rho_i = \rho_j$, then $c_i > c_j$ if $\gamma_i > \gamma_j$, (3) otherwise, one of them is selected randomly. Only top M_{sample} chromosomes are selected from the sample pool and stored in a *mating pool*. The mating pool contains a list of chromosomes, which are used for *crossover* and *mutation* described next.

Here we present an example of the selection process. Let consider the previous 3DTI setup for a TI conversation activity, where s_1^A, s_1^B and s_1^C are the audio streams and s_2^A, s_2^B and s_2^C are the upper body video streams. For a given chromosome, we can compute the average rate of audio streams $R_{s_{audio}}$ and average rates of video streams $R_{s_{video}}$, where $s_{audio} = \{s_1^A, s_1^B, s_1^C\}$ and $s_{video} = \{s_2^A, s_2^B, s_2^C\}$. Other cQoS parameters are average EED, average NVS and average NSC. We consider the rank with a 5-tuple $\rho = \{R_{s_{audio}}, R_{s_{video}}, EED, NVS, NSC\}$. We assume that for chromosome c_1 as shown in Figure 6 (d), the value of rank is ρ_1 = [64$kbps$, 2500$kbps$, 300ms, 2, 1] and for chromosome c_2 as shown in Figure 6 (e), the value of rank is ρ_2 = [256$kbps$, 1000$kbps$, 300ms, 2, 1]. According to the definitions of the TI conversation in Section 3.4, the priority ordering of the cQoS is $R_{s_{audio}} > R_{s_{video}} > EED > NVS > NSC$. Therefore, based on the rank definition above, c_2 has higher rank than c_1, since the value of higher priority cQoS parameter $R_{s_{audio}}$ is higher in c_2, even though the value of lower priority cQoS parameter $R_{s_{video}}$ is lower compared to c_1.

We consider another chromosome c_3 as shown in Figure 5(a) that has rank value ρ_3= [64$kbps$, 2500$kbps$, 300ms, 2, 1], which is equal to ρ_1. We can sort the chromosomes in order of their rank as $\{c_2, c_1, c_3\}$ (where c_1 and c_3 are ordered randomly since they have equal rank values). Based on the above definition, the crowding distances of c_2 and c_3 are ∞. The crowding distance for c_1 is $\gamma_1 = 192$. While performing selection between c_1 and c_3, since $\rho_1 = \rho_3$, we compare them using crowding distances. In this case, c_3 wins since $\infty > 192$. Therefore, if we are to select two chromosomes into the mating pool, we select c_2 and c_3.

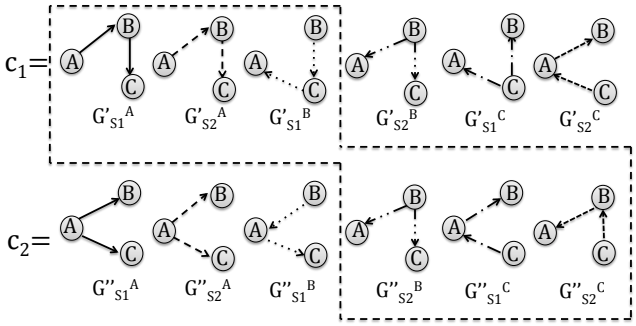

Figure 7: An example of crossover at the subgraph of stream s_2^B. The dotted box represents the offspring chromosome c'_1. The remaining parts create another offspring c'_2.

5.2.4 *Crossover*

We perform crossover between each pair of chromosomes in the mating pool. For crossover, we adopt the concept of common node [18]. Let us consider two chromosomes for the 3DTI setup used in Section 5.2.1, $c_1 = G'_{s_1^A}|G'_{s_2^A}|G'_{s_1^B}|G'_{s_2^B}|G'_{s_1^C}|G'_{s_2^C}$ and $c_2 = G''_{s_1^A}|G''_{s_2^A}|G''_{s_1^B}|G''_{s_2^B}|G''_{s_1^C}|G''_{s_2^C}$. A crossover is performed between them if a matching subgraph is found, i.e., $G'_s = G''_s$ for any $s \in \{s_1^A, s_2^A, s_1^B, s_2^B, s_1^C, s_2^C\}$. The crossover creates two offspring chromosomes by exchanging the distribution graph at the matching point. An example is shown in Figure 7. Since, $G'_{s_2^B} = G''_{s_2^B}$, a crossover is performed at the subgraph for s_2^B. The constructed offsprings are $c'_1 = G'_{s_1^A}|G'_{s_2^A}|G'_{s_1^B}|G''_{s_2^B}|G''_{s_1^C}|G''_{s_2^C}$ and $c'_2 = G''_{s_1^A}|G''_{s_2^A}|G''_{s_1^B}|G'_{s_2^B}|G'_{s_1^C}|G'_{s_2^C}$.

If subgraphs are matched at multiple locations, then a random location is considered for offspring construction. If the constructed offspring chromosomes satisfy the minimum cQoS quality con-

straints and bandwidth constraints, then they are inserted into the sample pool.

5.2.5 *Mutation*

To construct a mutant chromosome from the mating pool, we randomly pick a stream and replace the subgraph of that stream with another random distribution subgraph, which satisfies the minimum quality constraints and bandwidth constraints shown in (1) and (2) of the minimum quality problem. The algorithm for building the random subgraph is the same as the Algorithm 1.

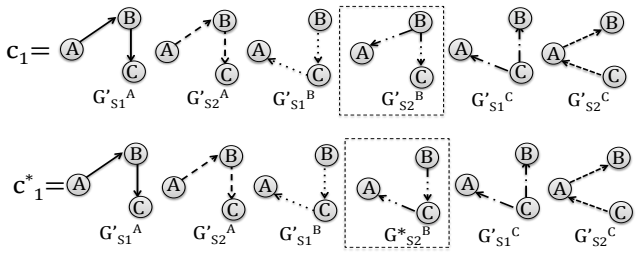

Figure 8: An example of mutation at the subgraph of stream s_2^B. $G'_{s_2^B}$ is replaced by a random graph $G^{\star}_{s_2^B}$.

An example of mutation is shown in Figure 8 over the chromosome c_1 of the previous example. We randomly pick the subgraph for stream s_2^B and replace the distribution graph $G'_{s_2^C}$ using another randomly populated distribution graph $G^{\star}_{s_2^B}$. It creates a mutant chromosome $c_1^{\star}$. Note that in all stages of the genetic operation, we only consider the graphs that meet the minimum quality requirements. The mutant chromosome is then added into the sample pool.

5.2.6 *Optimization Procedure*

The optimization process starts with the initial samples generated by ViewCast and moves towards the near optimal solution. The whole process is iterated for a fixed number of times (MG) or the consecutive L number of iterations that generates the same result. The steps in the optimization process are given below.

1. Set $i \leftarrow 0$; Construct a set of initial solution of size M_{sample} using ViewCast and add them into a sample pool.
2. Perform cQoS priority based selection over the chromosomes in the sample pool based on rank and crowding distance and create a mating pool of size M_{mating}.
3. Perform cross over between each pair in the mating pool and add the valid offsprings into the sample pool.
4. Perform mutation for each chromosome in the mating pool and add the valid mutant into the sample pool.
5. Select the best chromosome c from the sample pool based on rank and crowding distance. If c has been repeated for L consecutive times, return c, else keep count for c.
6. $i++$; If ($i < MG$) , Go to step 2, else, Select the best chromosome c from the sample pool and return c.

6. ILLUSTRATIVE EXAMPLES

The previous section provides a model for constructing content distribution graphs considering cQoS specifications and content demands. Here, we present two concrete examples with larger setup: one for TI conversation and the other for TI virtual lightsaber fight activities with four sites. The activity descriptions are given in section 3.4. All participants are using the same view orientation. Therefore, the priority ordering of the remote streams are equal for each participant. We make this assumption for the ease of explanation, however, violating this assumption does not require any changes in our algorithm.

6.1 Input Space

3DTI Setup. The list of input devices (each device is connected to an individual PC) connected at each participating site is shown in Figure 9 (a). The TI conversation session uses 1 audio and 4 video streams (two upper body streams and two lower body streams) from each site. On the other hand, the virtual lightsaber session generates an additional input sensory stream connected to the lightsaber, which indicates the position of the lightsaber in the virtual space. Table 3 shows the list of streams generated by the application setup.

Communication Graph. The complete network communication graph between the participating gateways $V = \{A, B, C, D\}$ is shown in Figure 9(b). Labels on the edges define the end-to-end delay (in millisecond) between gateways. Each vertex is labelled with a two-tuple, (ibw_v, obw_v) $(v \in V)$, which is inbound and outbound bandwidth capacity (in kbps) of the site.

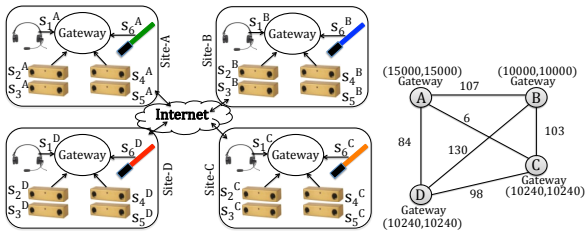

Figure 9: (a) TI application setup (showing only input devices), and (b) corresponding overlay communication graph showing end-to-end delay and available bandwidth.

cQoS Specifications. In the TI conversation, each participant requires at least 2 streams (audio and the highest priority upper body video) from each remote site (i.e., 6 remote streams), i.e., $NSC \geq 2$ and $NVS \geq 1$. However, in the virtual lightsaber, at least 4 streams (the highest upper body video, the highest lower body video, lightsaber sensor and audio) from each remote site (i.e., 12 remote streams) must be received by each participant to have a successful virtual lightsaber session, i.e., $NSC \geq 4$ and $NVS \geq 2$. The stream request for each participant and the priority of the streams are shown in the first and second rows of Table 3 for a given view. We use the notations introduced in Section 3.4. We consider 9 cQoS parameters with bounds and priorities described in Table 2.

6.2 Prioritized Evolutionary Optimization

Figure 10 shows an example of the iteration process in the OSM session optimization for TI conversation. Each chromosome is represented by 20 subgraphs; one for each stream (1 audio and 4 video streams from each site) in the current TI session. Since TI conversation does not use lightsaber sensory streams, no subgraph is generated for streams in s_{light}. Figure 10(a) and (b) show two chromosomes c_1 and c_2 constructed using ViewCast (Algorithm 1) as part of the generation of initial solutions for genetic iterations. Graphs with empty edges represent the dropped streams.

During crossover, two offsprings are generated by mixing c_1 and c_2 at the common subgraph position. Since $G'_{s_1^B} = G''_{s_1^B}$, a crossover is performed at the subgraph for s_1^B. Figure 10(c) shows one of the offspring chromosomes generated after crossover (dotted boxes represent the subgraphs coming from c_1 and solid boxes represent the subgraphs coming from c_2). A mutation is then performed at the subgraph $G''_{s_3^B}$ over the generated offspring. The mutant chromosome $c^{\star}$ is shown in Figure 10(d). Both the offspring and mutant chromosomes are added into the sample pool

Table 3: Application setup for TI activities

Property	TI conversation	TI virtual lightsaber
Stream Type	$s_{audio} = \{s_1^A, s_1^B, s_1^C, s_1^D\}$, $s_{ub\_H} = \{s_2^A, s_2^B, s_2^C, s_2^D\}$, $s_{ub\_L} = \{s_3^A, s_3^B, s_3^C, s_3^D\}$ $s_{lb\_H} = \{s_4^A, s_4^B, s_4^C, s_4^D\}$, $s_{lb\_L} = \{s_5^A, s_5^B, s_5^C, s_5^D\}$, $s_{light} = \{s_6^A, s_6^B, s_6^C, s_6^D\}$	
Stream Priority	$s_{audio} > s_{ub\_H} > s_{ub\_L} > s_{lb\_H} > s_{lb\_L}$	$s_{light} > s_{ub\_H} > s_{lb\_H} > s_{audio} > s_{ub\_L} > s_{lb\_L}$

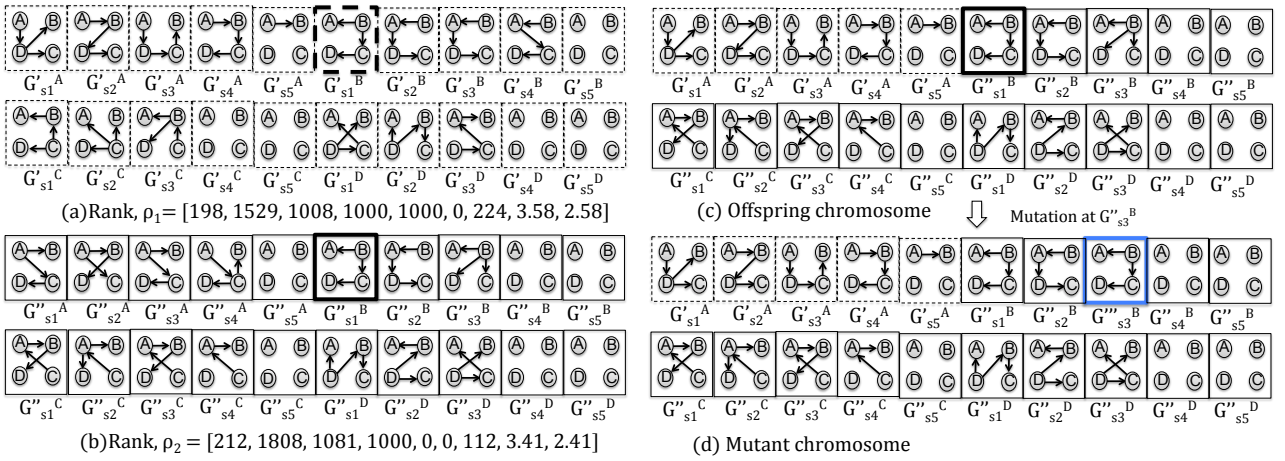

Figure 10: Optimization steps in TI conversation: (a) and (b) chromosome c_1 and chromosome c_2, respectively created using ViewCast, (c) offspring c_1' created using crossover between c_1 and c_2 at subgraph for s_1^B, and (d) mutant $c^\star$ created by replacing the subgraph for s_3^B in c_1' .

for the next iteration. Similar process can be shown for TI virtual lightsaber activity; however skipped due to the space limitation.

Here, the rank is a 9-tuple vector $\rho = [R_{s_{audio}}, R_{s_{ub\_H}}, R_{s_{ub\_L}}, R_{s_{lb\_H}}, R_{s_{lb\_L}}, R_{s_{light}}, EED, NSC, NVS]$. Though the computation of EED, NSC and NVS are straightforward from a given distribution graph, the allocation of rates for individual stream is tricky since the distribution graph is constructed considering the minimum bit rates of the streams. To allocate stream rates, after a distribution graph (or chromosome) is constructed with minimum rates, for each path in the graph, we allocate rates to the streams (associated to the path) in order of their priority (as shown in Table 3) subject to the bandwidth availability. For example, for c_1 in Figure 10(a), the directed edge e_{AD} is shared by four streams s_1^A, s_3^A, s_2^B and s_3^B. In our case, $s_1^A > s_2^B > s_3^A > s_3^B$. Initially, we assign minimum rates for each stream along the edge. The remaining bandwidth is then first assigned to s_1^A, if surplus is available after the maximum rate allocation to s_1^A, the remaining amount is assigned to s_2^B and so on. The rate allocation is performed from the higher priority streams to the lower priority streams.

Using the above approach, the ranks of c_1 and c_2 are computed as ρ_1 and ρ_2 and shown in Figure 10(a) and (b), respectively. Since, there is no light sensory stream in the TI conversation activity, the rate associated to it is zero. As c_2 assigns higher rate to the audio stream, due to the priority based comparison in the selection step (as explained in Section 5), c_2 wins over c_1.

We then run the optimization process iteratively using GA. Figure 11(a) shows the result after 2 iterations and Figure 11(b) shows the optimized solution generated after 20 iterations. The rank values are also given below the figures. Even the intermediate result (shown in Figure 11(a)) in GA provides higher rank compared to the initial solutions c_1 and c_2 (shown in Figure 10(a) and Figure 10(b), respectively). The optimized solution in Figure 11(b) assigns the the average rate of the audio stream to $256kbps$, and the average rate of the highest priority upper body video stream to $2500kbps$, which are the maximum possible in both cases. It also lowers the EED to $95.75ms$. However, the lower body video streams are completely dropped (i.e., low NVS and NSC). This

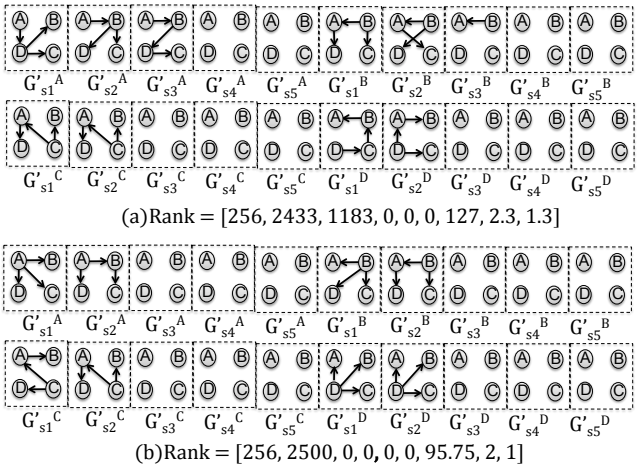

Figure 11: TI Conversation(a) after 2 iterations, and (b) optimized solution.

is acceptable since according to our activity definition, having only audio stream and upper body video streams from each site are enough for a successful TI conversation.

Figure 12 shows the optimized distribution graph for TI lightsaber activity with 24 streams (each site has 8 streams). It first optimizes the end-to-end delay (to $88ms$) and then the rate of lightsaber sensory stream (to $10kbps$). Streams s_{ub_H} and s_{ub_L} maintain the minimum quality ($\geq 1000kbps$). The rate of the audio streams are given a lower priority in optimization. If we compare the optimized rank values between TI conversation and TI lightsaber activities, it is evident that both the distribution graphs are optimized considering the requirements of the respective activities.

7. PERFORMANCE ANALYSIS

7.1 Simulation Setup

The simulator first generates a complete graph of an application-level overlay network. We use a mesh topology for the overlay net-

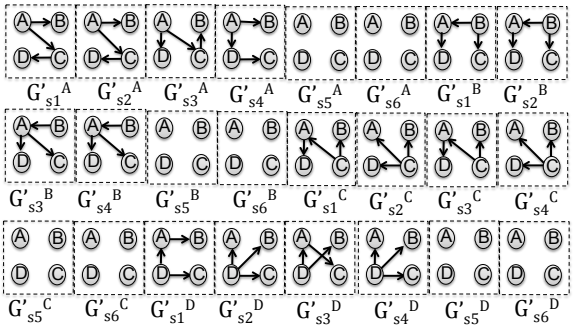

Figure 12: Optimized graph for TI lightsaber activity.

work, where the connectivity between the vertices follows a distribution collected from PlanetLab [21]. The total number of vertices, denoted as the *session size*, ranges from 3 to 10.

Figure 13: Geographical distribution of PlanetLab nodes.

The geographical distribution of the PlanetLab nodes used as vertices in our simulation is shown in Figure 13. We take vertices distributed widely over the world. Using a running service [9] from the PlanetLab, we collect outbound bandwidth information of each node (the lowest outbound we measure is 20000$kbps$). We assume that no other slices on the same node are competing for the bandwidth. For simplicity, we set the inbound bandwidth same as the outbound bandwidth measured. The end-to-end delays between the vertices are measured by running `ping` inside the nodes. The distribution of bandwidth as well as the end-to-end delay between the vertices are shown in Figure 14(a) and (b), respectively. X-axis of Figure 14(b) represents different communication edges. For 10 nodes, we have total 45 edges.

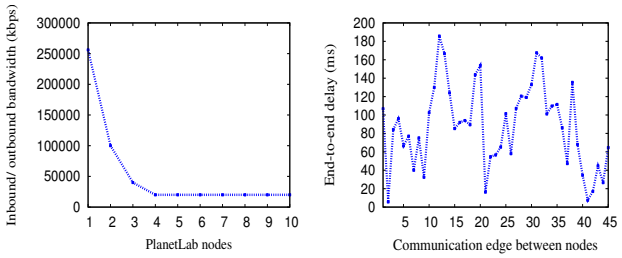

Figure 14: (a) Inbound/ outbound bandwidth distribution of 10 nodes, and (b) end-to-end delay between them. X-axis of (b) represents communication edges. For 10 nodes, we have total 45 edges.

In the experiment, each vertex has 16 video streams, which are evenly distributed in the 360^0 space; 8 covering the upper body and 8 covering the lower body of the participants, one audio stream and one light sensory stream. For any view request to a vertex, at most 8 of its original streams are selected for an optimal coverage of upper and lower body of 180^0 space along with the audio and sensory stream. The optimization process is implemented using java.

We consider two types of TI activities: conversation activity and virtual lightsaber fight activity as described in Section 6. The priority of the cQoS parameters for these TI activities and their minimum and maximum bounds are the same as described in Table 2. However, the number of streams and vertices is variable in our simulation. To have a successful conversation session, each vertex should receive at least one (the highest priority) upper body video stream and the audio stream from each other vertex. On the other hand, to have a successful virtual fight session, each vertex should receive at least two video streams (the highest priority upper body video stream and the highest priority lower body video stream), the audio stream and the light sensory streams from each remote vertex.

7.2 Performance of Session Optimization

To measure the performance of our optimization technique, we compare the rank of the distribution graph constructed by OSM with the rank of the graphs constructed by two other techniques: 1) ViewCast [32], and 2) stream-based random multicast [28]. In random multicast, the priority of the streams is not considered. Multicast trees are constructed for each stream by considering random ordering of vertices. We vary the number of nodes in our evaluation and each experiment was run one hundred times with randomly selected PlanetLab nodes from the set shown in Figure 13.

We run GA until 3 consecutive searches provide the same (best) result. The sample pool size in GA is set to 10 and the mating pool size is 5. The ranks are computed using the same technique shown in Section 6. For both activities, in this section, we only show the values of R_{audio}, R_{ub_H}, R_{lb_H}, EED, NVS, and NSC.

Figure 15(a)-(f) shows the values of rank parameters for different 3DTI setups. In case of TI conversation, the OSM provides the best audio (in Figure 15(a)) and upper body video (in Figure 15(b)) bit rate (hence, better quality) with the sacrifice of end-to-end delay (in Figure 15(d)), NSC (in Figure 15(e)) and NVS (in Figure 15(f)). Compared to the ViewCast, the maximum improvement in the audio bit rate is about 50% (for 7 vertices in Figure 15(a)) and compared to the random distribution topology, the maximum improvement is about 25% (for 6 vertices in Figure 15(a)). Since, for a successful conversation, lower body of the video streams are not required, in OSM, the rates of them are very low (in Figure 15(c)).

On the other hand, for TI virtual lightsaber activity, the end-to-end delay (in Figure 15(d)) is optimized along with the quality of the lightsaber sensory streams (not shown) by OSM. The quality of the video streams is balanced between the highest priority upper body video stream (Figure 15(b)) and the highest priority lower body video stream (Figure 15(c)) to construct a full body human image rather than trying to solely optimize the quality of only one portion of the body streams. Though the audio quality is poor (in Figure 15(a)) in this activity; it still maintains the minimum quality requirement.

As shown in Figure 15(e) and in Figure 15(f), the values of NVS and NSC are always higher in ViewCast for both TI conversation and TI virtual lightsaber activities. This is because ViewCast is originally designed to maximize the number of video streams in a TI session. However, both ViewCast and random distribution topology create higher end-to-end delay for TI virtual lightsaber and lower audio bit rate for TI conversation, compared to the OSM, since none of them consider prioritized cQoS based on the on-going activity requirements. Therefore, according to the definitions of the activities, OSM provides higher user satisfactions in both activities compared to the ViewCast and random multi-site and multi-graph distribution topology.

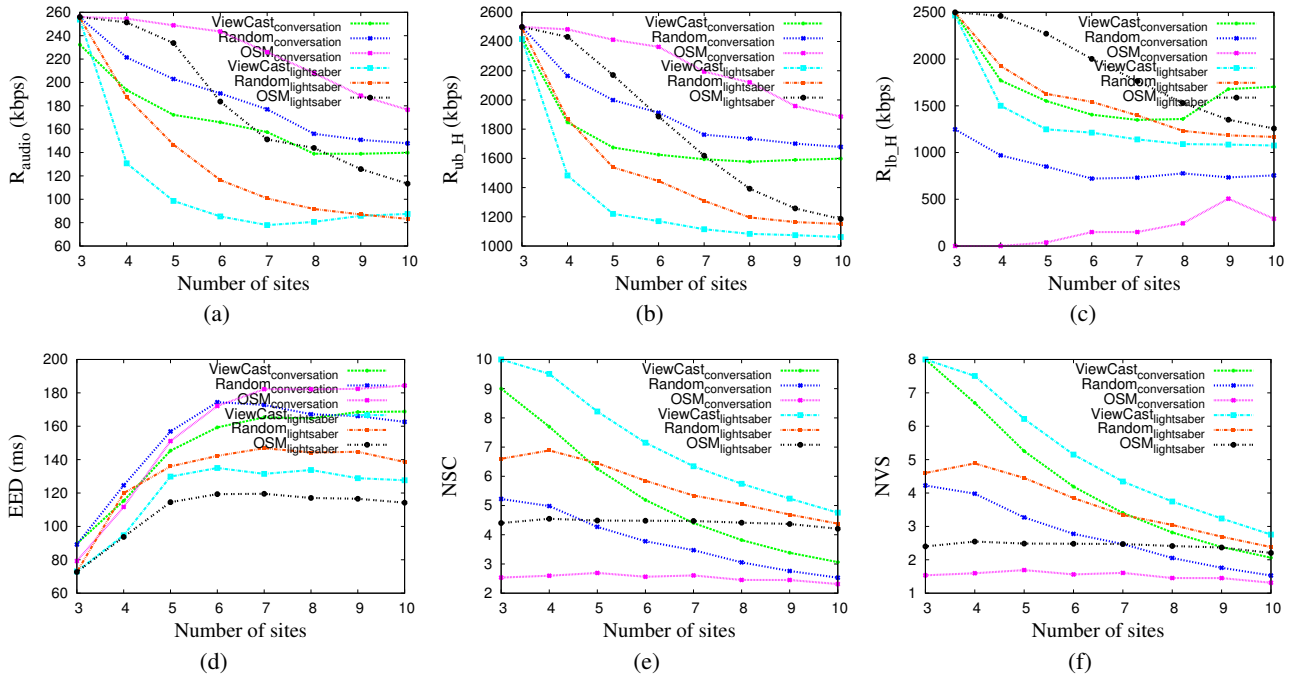

Figure 15: (a) Average rate of audio quality, (b) average rate of highest priority upper body video stream, (c) average rate of highest priority lower body video stream, (d) average end-to-end delay, (e) average number of streams received from each remote site, and (f) average number of video streams received from each remote site.

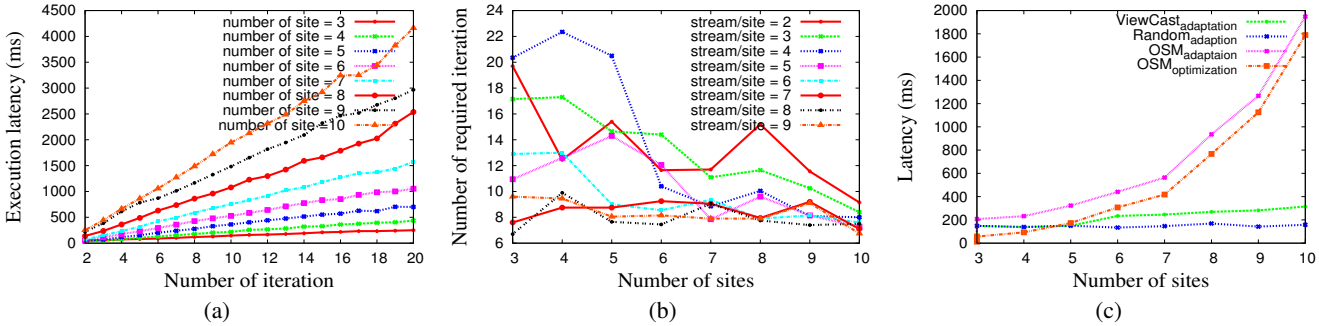

Figure 16: (a) Latency of session optimization for different number of iterations in OSM. (b) average number of iterations required to achieve near optimal solution in OSM, and (c) latency of session adaptation and session optimization.

7.3 Overhead of Session Optimization

In this section, we only use TI conversational activity, where each site can request maximum of 9 streams from each remote site.

7.3.1 Session Optimization Latency

A significant portion of the latency in OSM session optimization process comes from the iterations performed in the genetic operations (selection, crossover and mutation). Figure 16(a) shows the optimization latency for different number of iterations. This latency depends on the number of participating sites and the number of streams per site since the distribution graph becomes larger for large number of sites and large number of streams.

Figure 16(a) shows that the latency increases with the increase in the number of iterations. Though it takes around 4 seconds to perform 20 iterations for 10 sites, to generate a near optimal solution, the required number of iterations is much less than 20. To understand how many iterations are required to generate a near optimal solution, we plot the iteration count by varying the number of participating sites as well as the number of streams to request from each remote site in Figure 16(b). We consider a solution is near optimal when consecutive 5 iterations generate the same best solution for content distribution graph.

From the Figure 16(b), the number of iterations required for an optimal solution is higher for smaller number of participating sites as well as for smaller number of stream request. The reason is that having a small demand on the content distribution process creates a large solution space to search for an optimal solution. Because of this conflicting nature, most of the cases, the optimization latency is less than $500ms$ as shown in Figure 16(c).

7.3.2 Session Adaptation Latency

The session adaptation latency is the time a session layer takes to detect a session update, define cQoS specifications as a response, perform prioritized optimization and finally install session routing tables at the participants (same as the pipeline shown in Figure 3). We compare the performance of OSM with the performance of centralized topology constructions by ViewCast and random multicast. We run the experiments hundred times by varying the number of devices per site from 4 to 9.

Though the latency in OSM session adaptation is higher than ViewCast and random multicast, the process still takes less than two seconds. The sudden jump in the latency after 7 sites is due to the increase in the execution time for finding *valid offsprings* in the crossover as the demand becomes large. However, such latency can be masked using pre-computation of the optimization graph at the GSC by predicting the future changes in the users' views and activities.

7.3.3 CPU and Memory Overhead

Since OSM performs session optimization inside a centralized session controller, the overhead penalties of CPU and memory are of important concerns. In our experimentation, even with the 10 participating sites each requesting 9 streams from all other remote sites, the memory overhead is less than $100MB$, which is very low compared to the current off-the-self PC configurations.

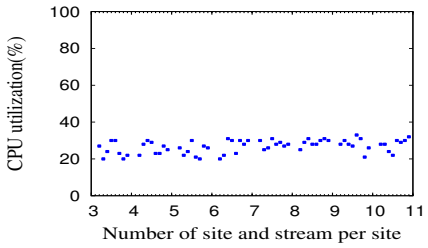

Figure 17: CPU overhead of OSM optimization process.

To measure the CPU overhead, we compute the CPU utilization of the optimization process by varying the number of participating sites and the number of input streams per site. The system configuration of GSC is: 2.8 GHz Intel Core $i7$ running Mac OS X (v. 10.7.4). The result is shown in Figure 17. The value $a.b$ in the x-axis represents the a number of participating sites each having b number of input streams.As the figure shows, the computation overhead is around $20\% - 30\%$, which increses very slowly with the increase of sites and streams.

8. CONCLUSION

This paper presents an evolutionary optimization technique for 3DTI session management considering the QoS specifications (minimum quality bounds and QoS priorities) defined by the 3DTI activity and content demands. We use three heuristics in GA to improve the 3DTI session optimization performance: 1) ViewCast-based generation of initial samples (solutions), 2) prioritized QoS-based sample selection, and 3) crowding distance-based sample variation. We show that our heuristics-based GA converges very fast and provides about 50% improvement in the allocation of desired QoS parameters. The process is computationally unobtrusive (in terms of CPU usage and computation time). We consider the cQoS specifications of TI conversation and virtual lightsaber gaming activities learned from our experiences. cQoS specifications for other 3DTI activities are considered for further investigations.

Acknowledgement. This work is supported by National Science Foundation grants NetSE 1012194, NetTS 0964081 & CSR 0834480. Any opinions expressed in this paper are those of the author(s) and do not necessarily reflect the views of the funding agency.

9. REFERENCES

[1] A. Arefin, Z. Huang and K. Nahrstedt et al. 4D TeleCast: towards large-scale dissemination of 3DTI contents. In *Proc of ICDCS*, 2012.

[2] A. Arefin, Z. Huang and R. Rivas et al. Tele-immersive gaming for everybody. In *Proc. of MM (demo paper)*, 2011.

[3] A. Arefin, Z. Huang and R. Rivas et al. Classification and analysis of 3D tele-immersive activities. In *IEEE Multimedia*, 2013.

[4] F. Bauer and A. Varma. Degree-constrained multicasting in point-to-point networks. In *Proc. of INFOCOM*, 1995.

[5] J. Cha, M. Eid, and A. Saddik. Touchable 3D video system. In *ACM TOMCCAP*, 2009.

[6] F. Chen, J. Liu, and Y. Zhao. Collaborative view synthesis for interactive multi-view video streaming. In *Proc. of NOSSDAV*, 2012.

[7] S. Chen and K. Nahrstedt. An overview of quality-of-service routing for the next generation high-speed networks: Problems and solutions. In *IEEE Network Magazine, Special Issue on Transmission and Distribution of Digital Video*, 1998.

[8] Cisco TelePresence- On-Stage Holographic Video Conferencing. http://www.musion.co.uk/ciscotelepresence, 2007.

[9] CoMon: A Monitoring Infrastructure for PlanetLab. http://comon.cs.princeton.edu/.

[10] D. Kumar, D. Kashyap and K. Mishra et al. Routing path determination using QoS metrics and priority based evolutionary optimization. In *Proc. of HPCC*, 2011.

[11] K. Deb. Multi-objective optimization using evolutionary algorithms. In *John Wiley and Sons*, 2010.

[12] F. Kuipers, P. Mieghem and T. Kokrmaz et al. An overview of constraint-based path selection algorithms for QoS routing. In *IEEE Communication Magazine*, 2002.

[13] F. Xiang, L. Junzhou and W. Jieyi et al. Qos routing based on genetic algorithm. In *Computer Communications*, 1999.

[14] M. Garey and D. Johnson. Computers and intractability: A guide to the theory of NP-completeness. In *Freeman*, 1979.

[15] H. Baker, N. Bhatti, D. Tanguay et al. Understanding performance in Coliseum, an immersive videoconferencing system. In *TOMCCAP*, 2005.

[16] M. Hosseini and N. Georganas. 3D videoconferencing application over an end system multicast protocol. In *Proc. of MM*, 2003.

[17] Z. Huang and K. Nahrstedt. Perception-based playout scheduling for high-quality real-time interactive multimedia. In *Proc. of INFOCOM*, 2012.

[18] K. Deb, S. Agarwal and A. Pratap et al. A fast and elitist multi objective genetic algorithms:NSGA-II. In *IEEE Transactions on Evolutionary Computation*, 2002.

[19] F. Kuipers and P. V. Mieghem. MAMCRA: a constrained-based multicast routing algorithm. In *Journal of Computer Communications*, 2002.

[20] D. Ott and K. Mayer-Patel. Coordinated multi-streaming for 3D tele-immersion. In *Proc. of MM*, 2004.

[21] PlanetLab. https://www.planet-lab.org/.

[22] R. Ravi, M.V. Marathe and S.S. Ravi ert al. Approximation algorithms for degree-constrained minimum cost network-design problems. In *Algorithmica*, 2001.

[23] S. Banerjee, C. Kommareddy and K. Kar et al. Construction of an efficient overlay multicast infrastructure for real-time applications. In *Proc. of INFOCOM*, 2003.

[24] The Perceptual Evaluation of Speech Quality. http://www.itu.int/rec/T-REC-P.862/en, 2001.

[25] S. Upadhyaya and G. Dhingra. Exploring issues for QoS based routing algorithm. In *Journal of Computer Science and Engineering*, 2010.

[26] Virtual Sword Fight. http://www.wired.com/gadgetlab /2012/07/smartphone-sword-fighting/, 2012.

[27] W. Wu, A. Arefin and R. Rivas et al. Quality of experience in distributed interactive multimedia environments: Toward a theoretical framework. In *Proc. of MM*, 2009.

[28] W. Wu, Z. Yang and I. Gupta et al. Towards multi-site collaboration in 3D tele-immersive environments. In *Proc. of ICDCS*, 2008.

[29] B. Wang and J. Hou. Multicast routing and its qos extension: problems, algorithms, and protocols. In *IEEE Network*, 2000.

[30] X. Chen, M. Chen and B. Li et al. Celerity: a low-delay multi-party conferencing solution. In *Proc. of MM*, 2011.

[31] X. Liu, J. Jiang and V. Sekar et al. A case for a coordinated internet video control plane. In *Proc of SIGCOMM*, 2012.

[32] Z. Yang, W. Wu, and K. Nahrstedt. Enabling multi-party 3D tele-immersive environments with ViewCast. In *TOMCCAP*, 2010.