TELKOMNIKA, Vol.15, No.1, March 2017, pp. 156~164
ISSN: 1693-6930, accredited **A** by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v15i1.3584

A Novel Clustering Tree-based Video lookup Strategy for Supporting VCR-like Operations in MANETs

Shijie Jia, Ye Guo*, Youzhong Ma, Chunlin Kuang, Guofang Kuang
Academy of Information Technology, Luoyang Normal University,
6# jiqing RD, Luoyang 471934, He'nan, China, (+86-379-68618331)
*Corresponding author, e-mail: 307806@qq.com

Abstract

Mobile Peer-to-Peer (MP2P) technologies can efficiently support large-scale deployment for video streaming services over mobile ad-hoc networks (MANETs) such as video-on-demand (VoD). Because streaming interactivity for VoD service causes frequent video lookup, the video lookup performance of MP2P-based VoD systems is significant for system quality of service (QoS) and user quality of experience (QoE). In this paper, we propose a novel Clustering Tree-based Video Lookup strategy for supporting VCR-like operations in MANETs (CTVL). CTVL builds the binary tree composed of chunks with high popularity by investigation of playback and switchover frequency of users for video chunks. Based on the built binary tree, CTVL designs a construction method of node community to group nodes into different communities in terms of cached chunks and defines the logical links between communities, which reduces maintenance cost of community structure and achieves high system scalability. CTVL also designs a video chunk lookup strategy, which makes use of the logical links between communities to achieve fast video chunk lookup. Extensive tests show how CTVL achieves much better performance results in comparison with other state of the art solutions.

Keywords: *binary tree, video chunk, community*

1. Introduction

Mobile ad hoc networks (MANETs) is a collection of randomly moving wireless devices within a particular area. In MANETs, each node acts as a router, forwarding data packets for other nodes without fixed base-stations to support routing and mobility management. MANETs can be used in failover systems, military and other applications and have attracted significant interest for numerous researchers [1]. As Figure 1 shows, provision rich-content to users for multimedia streaming services has been extremely popular digital business in the mobile Internet [2-4]. In particular, video streaming services have occupied the two thirds network traffic. The huge traffic demand requires efficient resouce sharing for the video streaming systems, which reduces the load of core network. By making use of the cooperation between peers, Peer-to-Peer (P2P) technologies can provide the feasible solution for the large-scale deployment of media streaming service. With the development of Mobile Internet, mobile Peer-to-Peer (MP2P) technologies have emerged as a state-of-the-art technology for video resource sharing in MANETs [5-8].

Video-on-Demand (VoD) services enable the playback point of viewers dynamically accessing any video content. VCR-like interactive operations lead to the change of playback point so that these operations bring two problems: matching supplier and seeking supplier. Matching supplier means that the requested playback content is located in the playback buffer of supplier. Seeking supplier means that the peer requesting new video content needs to know and connect with the supplier carrying the requested video content. The media server or peers in systems cannot bear the high maintenance cost for managing the peers carrying video resources. Moreover, the low efficient seeking supplier leads to the high lookup delay so as to influence the playback continuity. For instance, the "flooding" approach brings the large number of seeking messages so as to waste the valuable network bandwidth. Seeking supplier from discrete distribution of peers is challenging for supporting the VoD service.

Received October 17, 2016; Revised December 23, 2016; Accepted January 15, 2017

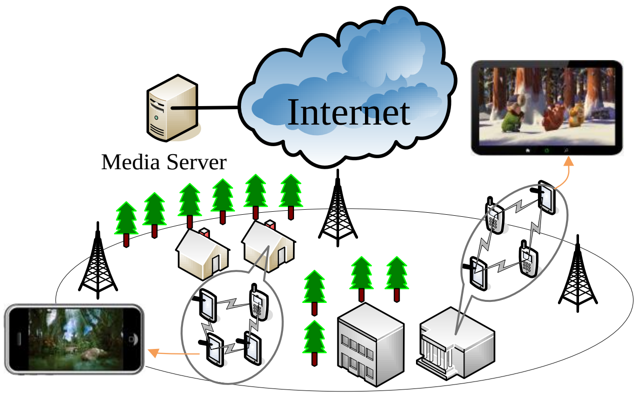

Figure 1.Video Streaming Services in MANETs

Numerous researchers have proposed some solutions for supporting the VoD services. For instance, SURFNet in [9] uses a hybrid AVL tree to group nodes in P2P networks, which supports fast resource searching. VMesh in [10] makes use of Distributed Hash Tables (DHT) to manage resources, so VMesh can improve the supply of video resources and support streaming interactive demands. However, the increase in the scale of P2P networks results in the high maintenance cost of structure, which will become the bottleneck of system's scalability. The optimal distribution of video resources with low-cost maintenance and balance between supply and demand is the key factor for the video sharing performance. The demand variation such as VCR-like operations continually breaks the balance of resource distribution, which results in the increase in the maintenance cost for the distribution. It is difficult to always require balanced distribution with low maintenance cost as there are many influence factors such as the node mobility except for the dynamic demand. Therefore, an efficient video lookup method adapting to the distribution variation also is the key factor for the video sharing performance.

In this paper, we propose a novel Clustering Tree-based Video Lookup strategy for supporting VCR-like operations in MANETs (CTVL) based on our previous work [11]. By estimation for popularity of video chunks and switchover of user playback point, CTVL builds the binary tree composed of chunks with the high popularity and groups the non-popular chunks to connect with the corresponding chunks in the binary tree. CTVL designs an estimation method of playback state stability of nodes and groups the nodes with stable playback state into communities corresponding to video chunksin the binary tree. The unstable nodes also are grouped into different communities in terms of the played video chunks. CTVL designs a video chunk lookup strategy to achieve fast video chunk lookup by making use of the built logical links between communities. Extensive tests show how CTVL achieves higher average lookup success rate, lower maintenance cost, lower average transmission delay and lower packet loss ratio (PLR) in comparison with other state of the art solutions.

2. Research Method

The media server is well-known to all mobile nodes and stores the original video resources. When the peers playing video content are unable to obtain the available streaming service from the overlay, it provides the streaming service for the requesting peers. Any video resource saved at the media server side is normally divided into *n* chunks with equal uniform length, namely $video=(c_1,c_2,...,c_n)$. Moreover, the media sever acts as the "entrance" of the mobile nodes joining system so that the cost of mobile nodes joining system is far less than "flooding". The media server maintains a node set where each item has joined the system based on the purpose of providing initial supplier for the new joining nodes in the future. The node set is defined as $NS=(n_1,n_2,\ldots,n_m)$. When a mobile node n_i expects to join the system, n_i sends a requested message to the server where the requested message includes the information of n_i and video chunk ID requested. The server adds n_i into *NS* and returns a response message containing the information of *NS* after receiving the request message.

n_i needs to select multiple nodes with the synchronous playback point relative to n_i from *NS* as initial candidate suppliers and contacts with them to obtain their current playback state.

The candidate suppliers return the response messages including their current playback state. If some candidate suppliers change their playback point or leave the system due to the streaming interactivity, n_i updates their state in *NS* and connects with the available supplier to obtain the streaming service. The media server is not responsible for maintaining the real-time status for each item in *NS.* If the nodes quit the system, they do not need to inform the server. This is due to the fact that the high maintenance cost for the state of items in *NS* will influence the scalability of system.

2.1. Binary Tree Construction of Video Chunks

The video chunk with the high popularity can attract the large number of viewers to access current chunk. The estimation of popularity depends on the statistics of access frequency for each video chunk. Let $logL=(log_1,log_2,\dots,log_m)$ be the log list stored in system. The access frequency of each chunk c_i can be defined as $f_i=\sum_{k=1}^{m} f_i^{log_k}$, where m is *logL*'s length and $f_i^{log_k}$ is the access frequency of video chunk c_i in any log k. The popularity of each chunk c_i can be defined as: $p_i = f_i / \sum_{c=1}^{n} f_c$. The mean value of total popularities of all video chunks is defined as $\overline{p}=\sum_{i=1}^{n} p_i / n$. If the popularity p_i of c_i is greater than $\overline{p}$, c_i is considered as the popular chunk. Otherwise, if $p_i \le \overline{p}$, c_i is considered as the non-popular chunk. All popular chunks form a set *PS* and all non-popular chunks also form a set *NPS*.

Except for the popularities of video chunks, the contact between chunks also is an important factor for clustering chunks. If the playback point of a node n_j jumps from c_e to c_i, c_e is considered as the fan-in connection chunk of c_i; c_i is considered as the fan-out connection chunk of c_e. The high frequency of fan-in and fan-out reflects the close contact between video chunks. Moreover, if a video chunk c_i has the fan-in and fan-out connections with many chunks, c_i is very important for the connection between video content. If c_i acts as "router chunk" (RC), the connections between c_i and contacted chunks can support fast movement of playback point.

Let $CN_{in}(c_i)$ and $CN_{out}(c_i)$ denote the number of fan-in and fan-out chunks of c_i, respectively. If $c_i \in PS$ and the value of $CN(c_i) = CN_{in}(c_i) + CN_{out}(c_i)$ is the largest among all items in *PS*, c_i becomes the root node and is removed from *PS*. The remaining chunks in *PS* form a new set $PS^{(2)}$. At the moment, c_i and video chunks which have the connections with c_i in *NPS* form a subset CS_i of video chunks. The non-popular video chunks in CS_i are removed from *NPS* and the remaining chunks in *NPS* form a new set $NPS^{(2)}$. If c_j is an items in $PS^{(2)}$ and the number of fan-in and fan-out chunks of c_j is the largest among all items in $PS^{(2)}$, c_j is considered as a RC. If c_j has the connections with c_i and c_j is the fan-in connection chunk of c_i, c_j becomes the left child node of c_i; If c_j is the fan-out connection chunk of c_i, c_j becomes the right child node of c_i. Otherwise, if c_j does not have the connections with c_i, the connection between c_i and c_j is built. At the moment, if $i > j$, c_j becomes the left child node of c_i; If $i < j$, c_j becomes the right child node of c_i. c_j is removed from $PS^{(2)}$ and the remaining chunks in $PS^{(2)}$ form a new set $PS^{(3)}$. c_j and video chunks which have the connections with c_j in $NPS^{(2)}$ form a subset CS_j of video chunks. The non-popular video chunks in CS_j are removed from $NPS^{(2)}$ and the remaining chunks in $NPS^{(2)}$ form a new set $NPS^{(3)}$. Similarly, the video chunk which has the largest number of fan-in and fan-out chunks in $PS^{(3)}$ is selected and becomes left or right child node of c_i. After iteration of the above process, a binary tree composed of popular video chunks can be built. The convergence condition of iteration is defined as: *NPS* and *NPS* are empty.

2.2. Construction and Maintenance of Node Community

The nodes with long online time can provide stable transmission of video data for the request nodes. Let $\bar{l}=\sum_{c=1}^{N} l_c / N$ denote the average online time in terms of the historical playback logs. N is the number of nodes in the historical playback logs and l denotes the average playback time of each node for each video chunk. When any node n_i finishes the playback of first chunk, n_i can be marked as the stable or unstable node in terms of average playback time of n_i. The average playback time of n_i is defined as $\bar{l}_i = \sum_{c=1}^{m} l_c(n_i)/m$, where m is the number of video chunks watched by n_i and $l_c(n_i)$ is the playback time for any chunk c. If $\bar{l}_i > \bar{l}$, n_i is a stable node; Otherwise, if $\bar{l}_i \le \bar{l}$, n_i is a unstable node. Any stable node needs to cache a video chunk in the binary tree into the static buffer. When the stable nodes quit the system, they remove the cached chunks. Initially, the system does not include any node. When a node becomes the stable node, it should preferentially cache the video chunk corresponding to the root node in the binary tree. The second and third stable nodes cache the video chunks corresponding to left and right child nodes of root node and they build the logical links with the first node. The subsequent stable nodes continue to cache other popular video chunks in terms of the breadth-first traverse and build the logical links with their parent nodes. The whole binary tree structure composed of stable nodes can support fast forwarding of request messages. When all video chunks in the binary tree are cached, the new stable nodes cache video chunks in the binary tree in terms of the popularities of video chunks. For instance, if each chunk in the binary tree is cached by a stable node at the time t_1 and the number of stable nodes is $SN(t_1)$, the node density corresponding to each chunk c_i in the binary tree is $1/SN(t_1)$. If $D_i=p_i-1/SN(t_1)$, $D_i >0$ and D_i is the largest among the difference values between popularities and node density of all chunks in the binary tree, the new stable nodes should preferentially cache c_i. The subsequent stable nodes cache popular video chunks according to the difference values between popularities and node density. All stable nodes which cache the same popular video chunks form a node community. The community member which has the longest average playback time becomes the head node in current community. Each node in any community contacts with the head node in current community. Moreover, each node in communities maintains the logical links with any member in communities corresponding to parent or child nodes in the binary tree, which ensures connectivity of binary tree composed of stable nodes.

On the other hand, the unstable nodes which cache any item in each subset CS_i form a node community. Any member in the community corresponding to CS_i maintains a logical links with a stable node in the community corresponding to c_i. The unstable nodes can make use of the logical links with stable nodes to search other video chunks by sending request messages. The stable nodes also make use of the logical links with unstable nodes to fetch information of suppliers cached other video chunks in CS_i, which serves other request nodes. The community corresponding to c_i has many stable nodes, so that the maintenance cost of links between stable and unstable nodes does not easily result in overload of stable nodes. The head nodes in all communities corresponding to the popular video chunks exchange the information including number of members in current communities.

n_i joins into the system and receives the data of video chunk from the supplier. The latter requires n_i to store information of a stable node n_p. If a node n_i finishes the playback for the first chunk and the average playback time of n_i is larger than $\bar{l}$, n_i becomes a stable node. When n_p which contacts with n_i finds that n_i can become a stable node by message exchange, n_p sends the information of n_i to the corresponding head node. The head node requires n_i to cache a video chunk into local static buffer in terms of the above caching rule. n_i joins into the node community corresponding to the video chunk cached in static buffer. When n_i finishes playback of the second chunk, n_i continues to exchange the current playback state (stable or unstable). If the average playback time is less than $\bar{l}$, n_i is a unstable node, removes the chunk cached in static buffer and quits the corresponding community. The stable nodes make use of the maintained links with stable nodes in other communities to implement fast resource lookup. However, the unstable nodes fetch the requested video chunks with the help of the contacted stable nodes, which increases the number of request forwarding. By the periodical estimation for the playback state, the role of nodes (stable and unstable) can achieve dynamic change. Although the stable nodes need to handle the request messages of the contacted unstable nodes, they can gain benefit from the lookup efficiency. Moreover, the large number of stable nodes can share responsibility for the load for handling request messages and maintaining state of the contacted nodes. Therefore, the node community structure does not frequent variation, so that the maintenance cost for community structure can keep low levels.

2.3. Video Chunk Lookup Strategy

After a node n_i joins the system, n_i obtains data of requested chunk $c_k \in CS_b$ from the selected supplier n_j. When n_j stores the information of n_i, n_i has joined the corresponding community. n_j sends information of the contacted stable node n_p to n_i. When n_i requests a new chunk c_h, it sends the request message to n_p. If $c_h \in CS_b$ and n_p is aware of the supplier cached c_h, n_p forwards the request message to the supplier; Otherwise, if n_p is unaware of the supplier cached c_h, n_p forwards the request message to the contacted head node. The head node broadcasts the request message in current community. When other stable nodes in the community are aware of the supplier cached c_h, they forward the request message. At the moment, n_i receives the data of c_h from the supplier and still is the member in the community corresponding to CS_b. If c_h is not included in CS_b and $c_h \in CS_a$, n_p forwards the request message to the contacted head node. The latter continues to forward the message to the head nodes in other communities corresponding to popular video chunks in the binary tree by making use of the built logical links. The request message also is forwarded to the corresponding supplier. At the moment, n_i quits the community corresponding to CS_b and joins into the new community corresponding to CS_a. n_i needs to remove the link with n_p and builds the new link with the stable node in the community corresponding to c_a. Obviously, the community structure based on the binary tree not only can speed up the lookup process (e.g. the requested and cached chunks belong to the same chunk subset) and reduce maintenance cost of logical links between nodes except for the lookup across chunk subsets.

3. Results and Analysis

3.1. Simulation Settings and Scenarios

CTVL is built in a MANET by making use of NS-2. CTVL was modeled and implemented in NS-2, as described in the previous sections. We choose a video clip of 400 *s* and evenly divide it into 20 segments, namely each segment has a fixed duration of 20 *s*. 400 mobile nodes are located in a range of x = 1000 *m* and y = 1000 *m*. The mobile speed range of nodes is between 0 and 30 *m/s*. The direction of each node is randomly assigned and the nodes' pause time is 0 *s*. 200 mobile nodes join the system following Poisson distribution and play video content in terms of 200 generated viewing logs. We also generate 10,000 historical playback logs to analyze the popularities and relationship of video chunks in order to build system structure of CTVL. The signal range of nodes is set to 200 *m*. The wireless routing protocol used is DSR. The default distance is set to 6 hops between the server and any node. The simulation time is 500 *s*. The bandwidth of the media server and each node are 10 *Mb/s* and 2 *Mb/s*, respectively. The rate and transport protocol of streaming data sent by any serving node or media server are 128 *kb/s* and UDP, respectively. The size of request message is set to 2 *KB*.

3.2. Performance Comparison and Analysis

The performance of CTVL is compared with that of SURFNet [9] in terms of average resource lookup success rate, maintenance cost, average transmission delay and average packet loss rate (PLR), respectively.

(1) *Average lookup success rate*: After the nodes send the request messages, the event that they successfully receive the desired video data from overlay network is considered as the successful lookup. The number of successful lookup divided by the total number of request is defined as the lookup successful rate.

As Figure 2 shows, SURFNet's blue curve has a fast rise trend from t = 50 *s* to t = 400 *s* and a slow increase from t = 450 *s* to t = 500 *s*. The red curve corresponding to CTVL's results also has a rapid increase from t = 50 *s* to t = 300 *s* and relatively slow rise from t = 350 *s* to t = 500 *s*. CTVL's increment and results are larger than those of SURFNet.

SURFNet investigates the similarity of stored content between nodes to build chain-based AVL tree structure. With the increase in the number of nodes, the overlay network has more and more available resources for the request nodes. However, the lookup strategy of SURFNet relies on the AVL construction, namely SURFNet needs to build a whole AVL tree to support forwarding request messages. Because SURFNet needs to investigate the online time of nodes to build AVL tree, the construction process of AVL tree needs to consume the large number of time, which leads to the low lookup success rate. Although CTVL also needs to investigate the playback time of nodes in the process of clustering nodes, CTVL only needs to compare the current playback time of nodes with average playback time of historical playback logs. Therefore, CTVL can fast build the tree structure to provide the resource lookup. Moreover, the resource lookup of SURFNet is subjected by the change of AVL tree, namely the state change of nodes in the tree seriously affected the resource lookup performance. This results in the frequent reconstruction of AVL tree. Therefore, CTVL can obtain higher average lookup success rate than that of SURFNet in the condition of limited available resources.

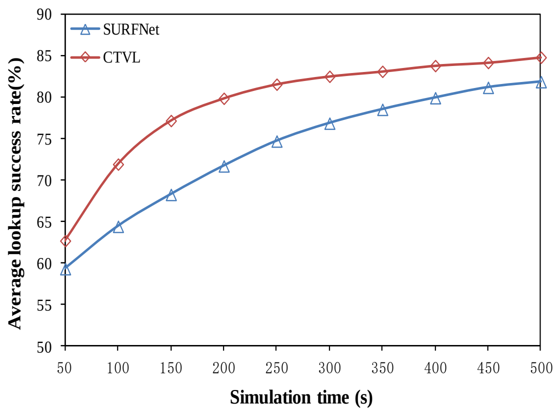

Figure 2. Average Lookup Success Rate Versus Simulation Time

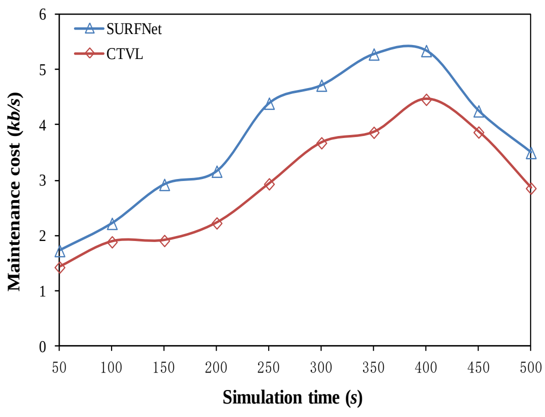

Figure 3. Maintenance Cost Versus Simulation Time

(2) *Maintenance cost*: The messages used by maintaining the overlay network such as nodes joining, leaving and lookup are considered control messages. The occupied bandwidth per second of control messages is defined as the maintenance cost.

As Figure 3 shows, SURFNet's results have a slow increase from *t* = 50 *s* to *t* = 200 *s* and fast rise from *t* = 250 *s* to *t* = 400 *s*. The blue curve also fast falls from *t* = 450 *s* to *t* = 500 *s*. The CTVL's red curve has a rise with low level from *t* = 50 *s* to *t* = 250 *s*, fast increases from *t* = 300 *s* to *t* = 400 *s* and rapidly decreases from *t* = 450 *s* to *t* = 500 *s*. CTVL's increment and peak value (close to 4.5 *kb/s*) are less than those of SURFNet (close to 5.4 *kb/s*).

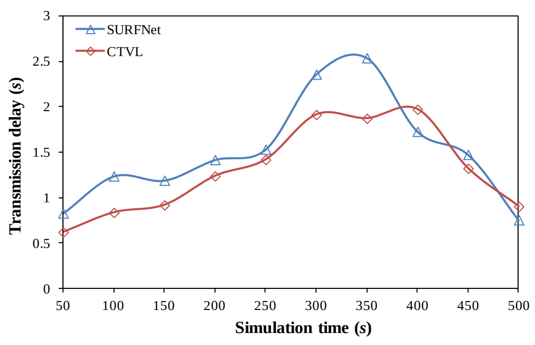

Figure 4. Average Transmission Delay Versus Simulation Time

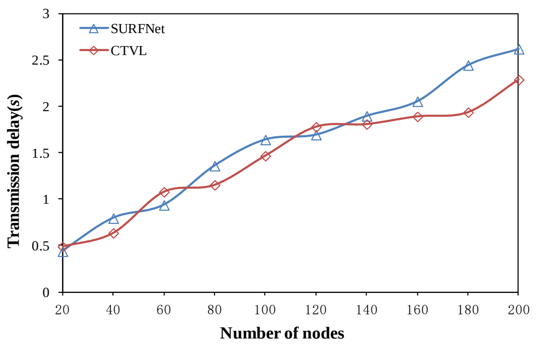

Figure 5. Average Transmission Delay Versus Number Of Nodes

SURFNet uses a chain-based AVL tree structure to group the nodes, namely SURFNet not only maintains the AVL tree structure, but also manages the nodes of chains. The node churn of tree and chains leads to reconstruction of overlay network, so that SURFNet needs to use large number of control messages to maintain the hybrid structure. With increasing number of nodes and request messages, SURFNet's maintenance cost fast increases. CTVL has a simple structure to group the nodes in overlay network. The autonomous maintenance strategy of nodes in communities can reduce maintenance cost. For instance, if the video chunks requested by the unstable nodes belong to the current chunk subset, the nodes do not need to implement the switchover between communities. Moreover, the flexible system structure can use low cost to address the state churn of nodes. CTVL can obtain lower maintenance cost than that of SURFNet.

(3) *Average transmission delay*: We calculate the total delay time of receiving data at the application layer. The total delay time divided by the amount of data received is used to indicate the average transmission delay.

Figure 4 shows the variation of average transmission delay during intervals 50 *s* of two solutions with increasing simulation time. SURFNet's delay curve has a fast rise with severe jitter before time *t* = 350 *s* and falls from *t* = 400 *s* to *t* = 500 *s*. SURFNet's results experience a severe congestion from *t* = 250 *s* to *t* = 400 *s*. CTVL's delay curve has the same shape with that of SURFNet, but CTVL's increment and jitter level are less than those of SURFNet. Moreover, the maximum delay experienced by SURFNet is 2.5 *s*, with 40% higher than that of CTVL (close to 2 *s*).

As Figure 5 shows, SURFNet's blue curve has a fast rise with severe jitter with increasing number of nodes and reaches a peak value 2.6 *s* at *t* = 400 *s*. CTVL's red curve also has fast rise trend during the whole process of node joining system, but CTVL's increment and peak value (close to 2.3 *s*) are less than those of SURFNet.

The transmission delay includes resource lookup delay and data transmission delay. In SURFNet, the resource lookup relies on the AVL tree, so the relatively low lookup success rate leads to the long transmission delay. For isntance, when the nodes cannot successfully search the supplier from overlay network, they only obtain the video data from the media server. There is 6 hops between nodes and server, which results in high lookup delay. CTVL has higher lookup success rate than that of SURFNet, so the nodes can obtain the video content from overlay network. Because SURFNet and CTVL do not consider the mobility of mobile nodes, the delivery of video data is severely influenced by the variation of geopolitical location of nodes. Therefore, the variation process of curves of SURFNet and CTVL is similar.

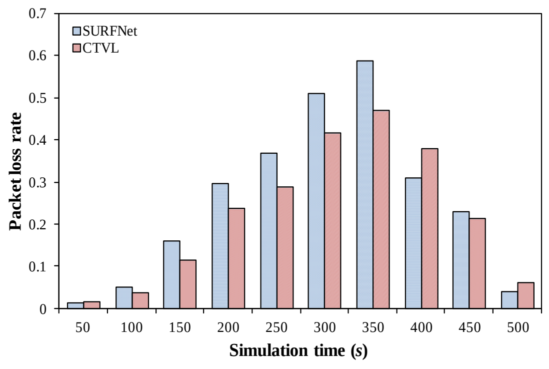

Figure 6. Average Packet Loss Rate Versus Simulation Time

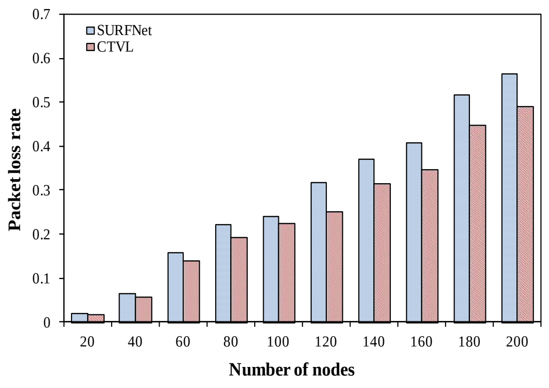

Figure 7. Average Packet Loss Rate Versus Number of Nodes

(4) *Average packet loss rate*: The packet loss rate is defined as the ratio between the number of packets loss and the total number of packets sent at the application layer.

Figure 6 illustrates the comparison between SURFNet and CTVL results for PLR with increasing simulation time. SURFNet's blue bars have fast increase from *t* = 50 *s* to *t* = 350 *s* and decrease from *t* = 400 *s* to *t* = 500 *s*. CTVL's red bars also experience a fall after fast rise during the whole simulation time, but CTVL's results have less increment and peak value (close to 0.47) than those (close to 0.59) of SURFNet.

As Figure 7 shows, SURFNet and CTVL are compared in terms of the PLR with the increase in the number of nodes. The results of SURFNet and CTVL have fast rise trend in the process of node joining system. However, CTVL has less increment than that of SURFNet.

Figure 7 can be seen clearly how CTVL has lower loss rate than SURFNet during the whole simulation.

SURFNet does not investigate the mobility of mobile nodes in the process of construction of hybrid structure, so that the nodes in overlay network relies on the logical link to search the suppliers. The mobility of nodes leads to the fast change of communication distance between nodes. For instance, when the two nodes have one-hop neighbor relationship at T_1, they may use the multi-hops transmission to deliver the video data at T_2 due to the mobility of nodes. Therefore, the transmission performance of SURFNet is difficult to adapt to the variation of communication distance, so that the PLR of SURFNet keeps higher level than that of CTVL. CTVL does not consider the mobility of community members in the process of clustering nodes, but the high-efficiency video sharing and low mainenance cost of CTVL reduce the network traffic. Therefore, the network congestion does not severely influence the video delivery performance of CTVL. CTVL's PLR is lower than that of SURFNet.

4. Conclusion

In this paper, we propose a novel Clustering Tree-based Video Lookup strategy for supporting VCR-like operations in MANETs (CTVL) based on our previous work [11]. By estimation for access frequency, the video chunks are divided into popular and non-popular chunks. The popular chunks form a binary tree; The non-popular chunks also are grouped in terms of the contacts between video chunks and connect with the popular chunks in the binary tree. CTVL designs an estimation method of node playback state in terms of playback time. The nodes are divided into stable and unstable nodes. The stable nodes cache the video chunks in the binary tree. The stable nodes are responsible for maintaining logical links with nodes which cache other chunks in the binary tree and forwarding request messages for video chunks. The unstable nodes also are grouped into communities in terms of the played video chunks and maintain the logical links with the stable nodes. The stable and unstable nodes make use of the built logical links between nodes to implement fast video chunk lookup. The stable playback state efficiently reduces the reconstruction frequency of tree structure, so the construction method of node community designed by CTVL can obtain low maintenance cost and high scalability. Moreover, the video chunk lookup strategy designed by CTVL relies on the built binary tree to achieve fast video chunk lookup performance. Extensive tests show how CTVL achieves higher average lookup success rate, lower maintenance cost, lower average transmission delay and lower packet loss ratio (PLR) in comparison with SURFNet.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant Nos. 61501216, 61272015, 61050004 and the Basic and Frontier Technology Research Project of Henan province (142300410303) and the Science and Technology Plan Projects (Openness & Cooperation) of Henan province (152106000048).

References

[1] Wang X, Li J. Improving the Network Lifetime of MANETs through Cooperative MAC Protocol Design. *IEEE Transactions on Parallel and Distributed Systems*. 2015; 426(4): 1010-1020.
[2] Xu C, Jia S, Wang M, Zhong L, Zhang H, Muntean G. Performance-Aware Mobile Community-Based VoD Streaming Over Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*. 2015; 64(3): 1201-1217.
[3] Jia S, Xu C, Guan J, Zhang H, Muntean G. A Novel Cooperative Content Fetching-Based Strategy to Increase the Quality of Video Delivery to Mobile Users in Wireless Networks. *IEEE Transactions on Broadcasting*. 2014; 60(2): 370-384.
[4] Sutjiadi R, Setyati E, Lim R. Adaptive Background Extraction for Video Based Traffic Counter Application Using Gaussian Mixture Models Algorithm. *TELKOMNIKA Telecommunication, Computing, Electronics and Control*. 2015; 13(3): 1006-1013.
[5] Bethanabhotla D, Caire G, Neely J. Adaptive Video Streaming for Wireless Networks with Multiple Users and Helpers. *IEEE Transactions on Communications*. 2015; 63(1): 268-285.
[6] Xu C, Zhao F, Guan J, Zhang H, Muntean G. QoE-driven User-centric VoD Services in Urban Multi-homed P2P-based Vehicular Networks. *IEEE Transactions on Vehicular Technology*. 2013; 62(5): 2273-2289.

[7] Shen H, Lin Y, Li J. A Social-Network-Aided Efficient Peer-to-Peer Live Streaming System. *IEEE/ACM Transactions on Networking*. 2015; 23(3): 987-1000.
[8] Xu C, Jia S, Zhong L, Muntean G. Socially aware mobile peer-to-peer communications for communitymultimedia streaming services. *IEEE Communications Magazine*. 2015; 53(10): 150-156.
[9] Wang D, Yeo C. Superchunk-Based Efficient Search in P2P-VoD System Multimedia. *IEEE Transactions on Multimedia*. 2011; 13(2): 376-387.
[10] Yiu W, Jin X, Chan S. VMesh: Distributed Segment Storage for Peer-to-Peer Interactive Video Streaming. *IEEE Journal on Selected Areas in Communications*. 2007; 25(9): 1717-1731.
[11] Jia S, Xu C, Guan J, Zhang H. *A MP2P-based VoD Solution for Supporting VCR-likeOperations in MANETs*. Proceedings of IEEE International Conference on Cloud Computing and Intelligence Systems. Hangzhou. 2012: 762-766.