A Dynamic Hybrid Service Overlay Network for Service Compositions

Yousif Al Ridhawi[1]**, Gajaruban Kandavanam**[1]**, and Ahmed Karmouch**[1]
[1]Department of Electrical and Computer Engineering, University of Ottawa, Ottawa, Ontario, Canada

Abstract - *This paper presents a dynamic hybrid Service Overlay Network (SON) for service composition for multimedia delivery in mobile networks. The overlay considers the nomadic nature of mobile nodes in the decision for node placement within the overlay as well as the types of services provided and expected Quality of Service (QoS) levels. Dynamic re-organization of the overlay reflects changes in stability and QoS provided by the overlay nodes, thus promoting nodes providing high QoS towards the center and demoting unstable nodes towards the edge of the network. This paper also presents a mechanism to efficiently search for services based on the type of service and the required QoS to meet client's expectations.*

Keywords: hybrid overlay, SSON, QoS, service composition

1 Introduction

Delivering customized media by providing advanced services through service composition and forming Service Specific Overlay Networks (SSONs) requires dynamic construction of service-specific overlays without prior knowledge of the underlying physical network. These media services include dropping frames to meet QoS constraints, caching media content before being viewed and converting media formats to meet the clients' needs. The process becomes more challenging in mobile networks where nodes are not fixed, and may arrive and depart from the network without warning thus increasing the topology's instability.

Establishing SSONs involves discovering network-side nodes that support required media processing capabilities, deciding which of the discovered nodes should be included, and configuring overlay nodes [1]. Using a registry-like solution for information has many limitations, such as scalability, difficulty and cost involved in keeping an up-to-date registry in a dynamic environment, and single point of failure. The idea behind SSONs emerged from SONs [2]; application-layer logical networks formed by links between service nodes with matching inputs and outputs. The initial proposal faced many problems with how complex service interactions should be dealt with in mobile environments. In the Ambient Networks architecture [3], SMART (Smart Multimedia Routing and Transport Architecture) [4], was proposed for guiding media flow through specialized network nodes to make use of their ability to cache, transcode, synchronize multimedia data or any other service through the use of overlay networks in an Overlay Control Space (OCS). OCS is responsible for selecting the necessary media processing nodes and establishing a service-specific end-to-end overlay network for media delivery.

SSONs are composed of three types of nodes: Media Ports (MP), Media Servers (MS), and Media Clients (MC). MPs are nodes supporting special services such as caching, synchronizing and media transformation. MCs are the actual clients that request media services. MSs are sources of media flows from which MCs receive the desired media services. SSONs are built to provide composite services based on technical, QoS, and Quality of Experience (QoE) requirements of MCs for basic services originating from MSs. This is done through a process consisting of a set of sub-processes between the MC and the MS. Each sub-process is performed by a service provided by an MP. This composition can be specified during design time or discovered at runtime. The former has many limitations especially in mobile environments, where service nodes are in constant movement and would face recovery and reconfiguration difficulties when there are node failures or visible degradation in the QoS. The latter approach however can maintain high levels of QoS during execution of composite services and has the ability to replace component services for changes in the requirements.

The contribution of this paper is to present a multi-layered approach to building SSONs based on predefined QoS. This is done by establishing a lower hybrid overlay network connecting MP nodes. In this proposed overlay, nodes of similar services are clustered in well-known regions. Additionally, the nodes are organized such that those with higher QoS are located closer to the center of the MP space (hybrid overlay) while those of lower quality are dispersed further away from the center. This method improves response time, reduces overhead and number of overlay hop counts required to provide the requested services. This is done in a way the network's resiliency to dynamic mobile topologies is improved. In such a network, overlay nodes, and their respective underlay mobile service nodes are constantly in motion with unpredicted arrivals and departures.

The remainder of this paper is organized as follows. Section 2 presents the related work. Section 3 presents an

overview of a proposed hybrid MP overlay network. Section 4 describes a score-based overlay management method. Section 5 our circular MP search for service compositions. Section 6 discusses our simulation and system evaluation. Finally, section 7 provides our conclusion and future work.

2 Related Work

Gou et al. [5] presented a hybrid Peer-to-Peer (P2P) Overlay network for highly efficient searches by dividing the overlay into structured super peers connected through a ring-like structure, and standard unstructured peers forming an optimal tree connected to a super peer. Queries in the hybrid structure have a maximum number of hops after which the query is dropped. A major drawback of this method is the inverse relationship between the overlay's re-configurability and resource availability seen in the breadth and depth of the optimal tree each of these super nodes forms. In real networks, resource limitations in super peers require placing limitations on the number of direct regular peers connected as well as how deep trees can grow. Consequently, a failure of a high-level peer in a tree may require costly reconfigurations. Additionally, the proposed work did not illustrate how failures should be handled to minimize information loss, or message overhead for network reconfiguration and stabilization.

Han et al. [6] presented a solution to achieve a sustained throughput and fast data dissemination rate from any data source with a goal of minimizing overhead even under frequent node failures through a Churn-Resilient Protocol (CRP). The Chord-like structure proposed dynamic fingers among nodes satisfying certain proximity properties. From the CRP overlay, a shared spanning tree is extracted with a minimized latency. In the first hops of the tree, scope-flooding is used and tree traversing in the remaining hops.

The authors correctly indicated tree-based dissemination methods are vulnerable to node failures. Hence they proposed the use of churn resilient overlay nodes to achieve good fault resilience. New nodes form finger links to other nodes using proximity measured through latency and capacity thus placing high-capacity nodes at high positions in the delivery tree. The purpose of choosing a Chord-like structure is to overcome the vulnerability of tree structures to node failures. Any node failure can be temporarily bypassed through links neighboring nodes have to other Chord nodes in the overlay. Hence if any node fails with probability $e^{-a(a>1)}$, the structure is connected with a probability higher than $1\text{-}n^{1-a}$- A link between any two nodes is given a network proximity weight measured by the link's latency and capacity difference between the two nodes. Although the solution improved traditional Chord networks, it's reliance on structured P2P overlay meant data placement and topologies are tightly controlled. Although structured P2P solutions have efficient data access and reduced number of search hops, processing fuzzy queries is challenging. In addition, the stabilization costs associated with the arrival and departure of nodes exceeds the original cost of Chord.

Chord was presented as a distributed lookup protocol that efficiently located nodes with particular data [7]. This was implemented through DHTs that associated a key with each data item. Our proposed lower overlay utilizes Chord structures for their scalable join and leave process, ability to provide an organized structure for fast search of nodes within predetermined regions of the overlay, and ability to provide correct lookup results even during recovery.

To reduce search time for available services in P2P systems, Li et al. [8] illustrated a super-chunk-based fast search network (SURFNet) for prompt Video on Demand (VoD) delivery. Their design had some limitations; need for timely content location and delivery, and limited cache size. Therefore, multimedia files were divided into chunks and grouped into super-chunks. Peers formed a structured overlay based on the super-chunks they held. This structure consisted of an AVL tree layer of stable peers and a holder chain layer of nodes holding the same super-chunk. All remainder peers formed an unstructured overlay that periodically exchanged chunk-level data availability information with neighbor nodes.

Makaya et al. [9] focused on user-generated contents and user-generated services in Next-Generation SONs (NGSON). The objective was to support efficient delivery of services while being end-user-centric by allowing network operators and service providers to have service management, control, creation, composition, and execution. However, the approach was highly centralized requiring an NGSON controller to act as an overlay manager. The controller formed a primary point of contact for end-users and service nodes responsible for service composition, orchestrating, chaining, service node discovery and monitoring, overlay management for self-organization and sessions and path establishment between service nodes. However, the approach faced the problem of a central point of failure, and lack of scalability. Our work focuses on finding a decentralized solution for service composition that maximizes the benefits and minimizes the limitations found in centralized and decentralized solutions.

3 Media Port Hybrid Lower Overlay

We focus our work on dividing the Overlay Network layer in the SMART SSON architecture into two layers; a lower overlay and the SSONs built over it. The lower overlay constitutes a hybrid overlay structure whose node organization and links reflect the types of services provided by the nodes, their stability, and QoS associated with services provided by them. The lower overlay network is composed of all MP nodes present that can be utilized during formation of a service composition between the MS and the MC. MPs are chosen to form SSONs reflecting the MC's needs in terms of QoS and expected QoE. By using a lower overlay network, we are able to organize MPs prior to any service requests from MCs for fast MP search and SSON construction. Nodes are assumed to be mobile, hence failures are expected.

Promoting use of hybrid overlay stems from the following:

• Searching for component services to form a composite service according to an expected level of QoS may require a large number of search messages exchange between nodes.
• When forming SSONs, it is beneficial to find MPs in the same path from MS to MC, provided required QoS for the service is available and the MPs have low hopcount reflecting low number of hops in the network level thus reducing delay.

3.1 Justifying the use of the Hybrid Overlay

Our proposed hybrid structure forming our lower overlay network is used to overcome limitations seen in both structured and unstructured overlay networks. Unstructured overlay networks are easily constructed with arbitrary overlay links. However, there is notable limitation when a peer is interested in acquiring some data. In this case, the query needs to be flooded in the network to find a node holding the desired data. The less popular the service clients are interested in acquiring, the less likely it will be found in a nearby node to the requestor. Consequently, the number of messages exchanged would consume significant bandwidth in a wireless network besides the operational overhead. In addition, since no correlation exists between service requested and corresponding MP, it is highly unlikely that a search is successful. Structured overlay networks overcome many of the limitations of unstructured networks by maintaining DHTs and distributing content in the network. The disadvantage of structured overlays is the large number of messages that need to be exchanged to stabilize the network, maintain existing links, and heal links to failing nodes that drop from the network unexpectedly. This rigidity is highly unfavorable in dynamic mobile environments where nodes leave the network without warning and new nodes are constantly arriving.

Hence, we propose the use of a hybrid overlay structure that maximizes the advantages of both structured and unstructured overlays and minimizes their disadvantages. Our overlay consists of several hierarchical layers of Chord-like rings and unstructured leaf connections. Each node in the network can be one of two types; a *major node* or a *minor node*. A Major node is any node capable of joining one of the Chord structures present within the network. These nodes should have the following characteristics:

• Have a long life span and are highly likely to remain in the network. This is directly related to the node's arrival time.
• Have adequate level of available resources to support links to nodes present within the same Chord structure as theirs, be the managing node for further sub-Chord structures and be the parent for other leaf (Minor) nodes.
• Have popular services frequently requested during service compositions.

Major nodes are divided into two types;

• *Normal Major Node (NM)*: a major node that may have any number of child leaf nodes, Chord nodes, and neighboring nodes. NM nodes do not link directly to higher Chord nodes.
• *Link Major Node (LM)*:a major node having the same characteristics as NM nodes with the addition of a link to a higher level Chord node.

A *minor node (leaf)* is any node linked to only a single node that is its parent and shares the following characteristics:

• Newly arrived nodes that have not built stability credentials. We assume they are short-lasting service nodes.
• Nodes that do not have the resources to support links to other neighboring nodes to be part of a Chord ring.
• Have unpopular services rarely used in compositions.

The formation of the hybrid overlay network is performed in three stages: initialization, sub-Chord formation and extension, and promotion-based stabilization and overlay management. Initialization stage occurs when the first '*m*' nodes start arriving forming the central Chord structure. Nodes continue to join Chord until C_{Thresh} threshold is reached; a network dependent variable related to the expected number of nodes in the network, available bandwidth and other criteria. Arriving nodes send a join request message to a nearby node chosen randomly and await a response message that includes the node's successors and finger links. If a response is received then the node joins the Chord Structure. Otherwise, the process is repeated and the join message is sent to another Chord node until an acceptance is received. Once a C_{Thresh} number of nodes have joined, the central Chord network is formed and an arriving node must request a join from one of existing Chord nodes. Receiving an acceptance, the node is added as a *minor node* to the responding node.

4 Score-based Overlay Management

To maintain a stable Chord network, nodes exchange messages to monitor departure of neighboring nodes and fix broken links. To maintain the dynamic form of our hybrid overlay we have added a new metric that nodes must inform their neighbors of, reflecting node's stability and supported QoS. The score for each node can be evaluated as follows:

$$NodeScore = [w_{arriv}(T_{curr} - T_{arriv}) + w_{serv}\sum_{i=1}^{n} P_i(\sum_{j=1}^{m} w_{Qual_j} Q_j)] \quad (1)$$

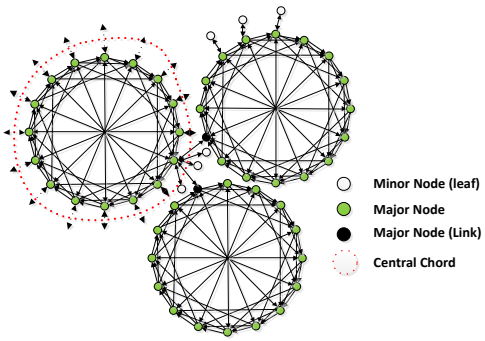

Figure 1. Proposed Hybrid Lower Overlay Network.

Where, w_{arriv} is the weight given to the node's total time within the system since the last failure. The longer a node remains in the system, the higher is its score, recognizing it as a stable node. Each MP can provide up to 'n' services ranging in P_i popularity levels reflecting the frequency with which MCs request these services. Additionally, each service can have a number of QoS criteria Q_j, such as delay and accuracy. Each of these is given a weight reflecting its importance, given by w_{Qualj}. Each Chord ring in the overlay has a minimum threshold, $MinThresh_i$. When a node's score falls below $MinThresh_i$, it is demoted to a lower Chord ring or becomes a *minor node* to neighboring Chord node. Similarly, if a *minor node*'s score exceeds $MinThresh_i$ then it can be promoted by a *major node* to join the Chord ring. Additionally, if a MP's score in any Chord ring exceeds a maximum threshold $MaxThresh_i$ set for that ring, then the node can be promoted to a higher level ring. In the event that no higher level Chord ring exists, the node will begin forming a new higher level Chord ring to become the new center of the lower overlay network. Fig. 1 illustrates a simplified view of hybrid lower overlay network showing the presence of two main types of nodes in the system; *major nodes* and *minor nodes*.

4.1 Overlay Join

When an MP arrives in the network, a join request must be sent to one of the *major nodes*. The receiving node evaluates the score of the arriving node to determine if an acceptance should be sent. If the joining node's score falls below the structure's *MinThresh*, then the request is rejected. Prior to rejection, the receiving node determines whether a lower Chord structure with a lower threshold exists. If such a structure exists, the request is forwarded to the *link major node* of that lower Chord. The *link node* may decide to accept the join request, forward it to one of its neighbor nodes or pass it to a lower Chord structure. If there is no lower level node to forward the request to, the receiving node may accept incoming node as a *minor node* or send a join rejection. If the node's score falls between the Chord structure's *Max*- and *Min-threshold*, then the node may either accept the joining node as one of its *minor nodes* or allow it to join as a neighbor within the same Chord structure. Finally if the node's score exceeds the structure's *MaxThresh*, the join request may be accepted and the node added as a *minor node* or as a neighbor in the Chord structure. Additionally, the request may be forwarded to a higher leveled Chord structure closer to the center of the MP space. The join process must be done in a time frame, $t_{response}$. Each node can only sustain a connection to a certain number of neighboring *major* and *minor nodes* depending on the available resources and QoS level that permits maintaining a high node score. Once the maximum number of nodes is reached, join requests received are rejected, forwarded to some other node or accepted after making modifications to its existing links to *minor nodes*.

4.2 Minor Nodes Reorganization

One possible approach to accepting new node links or reducing resource drainage on a *major node* is through *minor node* reorganization. The process is triggered by a *major node* contacting its *minor nodes* that are not become part of a lower level Chord structure and determining the node with the highest score. A *FormNewPartition* call is sent to that node informing it to start forming a new Chord structure. This node will become a *major link node* linking the parent node with the new lower level Chord structure and will maintain its status as a *minor node* to the parent node. A *JoinNewPartition* call message is sent to all other *minor nodes* informing them of joining the new Chord structure. The message includes the identity of the node they must contact; i.e. the sender node of the *FormNewParition* call message. The next step involves sending a join acceptance message to the join request sender node. The *major node* continues to use this same process of grouping its single *minor nodes* into lower Chord structures until all *minor nodes* have become *major link nodes* and is unable to accept any new join requests.

4.3 Handling Node Failures

4.3.1 Failure of Minor Nodes

Since each *minor node* is only linked to its parent *major node*, failure of *minor nodes* has no negative effect on the overlay structure. Affected *major node* simply removes any reference in its routing table to the failed node and can now accept an additional join request from an arriving MP.

4.3.2 Failure of Major Link Nodes

Failure of a *link node* disconnects its personal Chord structure from the parent *major node* and breaks the uniform shape of the Chord structure. Reconnecting the network has two phases. If the child node is a *major link node*, then failure of the parent node means separation of the lower level Chord structure and all attached sub-structures from the higher Chord structure that contained the parent node. In the first phase, the Chord structure detects failure of one of its nodes. All affected Chord nodes reconfigure their links to restructure the Chord network and form all links necessary to stabilize the network. In the second phase, from the level of the highest detached Chord structure, one node must be chosen as the new Major Link node. This node will connect with the same major node the old link node was connected. The process of choosing a new *link node* is done by a search process performed to find the node with the highest score. During lifetime of a Chord network, nodes periodically exchange stabilization messages to update successor, predecessor, and finger table links. We utilize these messages and expand them to include updates for node scores. Each node part of the optimal Chord structure maintains a table that includes node scores for its immediate successor, predecessor, and two finger links. Therefore, to discover the node with the highest score it would take at most *log n* messages. This process starts at any random node in the reconnected Chord structure. The node evaluates from the four nodes it is connected to as well as itself to find which of these nodes has the highest

score. A message is sent to the node's successor containing identity of the chosen node and the identity of the rejected ones. This node then performs the same process and passes the message to its immediate successor. This process continues until all nodes within the Chord structure or one of the nodes connected to it is informed. The node with the highest score is then informed of being chosen as the new *major link node*. The chosen node then sends a join request message to the previous *link node*'s parent to reconnect the detached Chord and its sub-structures.

5 Circular Media Port Space

In a work presented in [10], search for MPs to form a composition was performed based on three assumptions: The location of the MC is known, the location of the MS is known, and the search for MPs is performed in the direction emanating from the MC towards the MS only. Using this, any node receiving a query will forward the query to local nodes only if it is within α degree. The angle α represents the maximum search scope for possible component services. This value depends on the locations of MC and MS. This approach, in general, avoids sending queries to regions of the network where answers are not likely to be found. However, in wireless mobile networks, the dynamic movement of nodes at the network layer may result in an unequal distribution of MPs at the overlay level. We may end up with high density regions where large numbers of diverse MPs exist while other regions may lack any MPs or have them sparsely located. In the latter, it is highly likely that α must be continuously increased until all component services are found.

In our work, we overcome the limitations of the previous solution using two methods. The first is by providing a hybrid lower overlay structure discussed earlier. The second is by dynamic node promotion and demotion according to Eqn. 1. These two methods result in a circular MP space that is further re-organized based on the types of services provided. As a result, nodes arriving in the network must first determine the overlay region they must join. We assume the following:

- The center of the MP space is a known region that can be located by all nodes joining the network.
- Angles α and β representing the start and end borders, respectively, of each service-type sector are also well known by all nodes joining the network. α is measured from the absolute 0 degree angle pointing to a relative direction accepted by all nodes in MP space.

Using these assumptions, any search for a specific type of service forming a composition can be done by searching within specified regions in the MP space. A simple example may involve a video format conversion service that converts from MP4 to AVI. The MP space sector providing this service is to be located. For e.g., starting at $\alpha=47°$ away from the absolute $0°$ angle of the MP space, with a total angular range of $\beta=35°$. This region represents the total space within which this type of service may exist regardless of the QoS required to form the composition and meet the MC's QoE or the stability of the node. Stability in our case refers to the length of time the node has been present in the network and thus the length of time this node is likely to remain in the future. However, to further reduce the physical search area for potential component services we have indicated that the dynamic promotion and demotion of overlay nodes in the lower overlay network results in movement of nodes with higher scores towards the center of the network. Therefore, the nodes with higher stability, QoS, available resources, etc. are generally located closer to the center of the MP space.

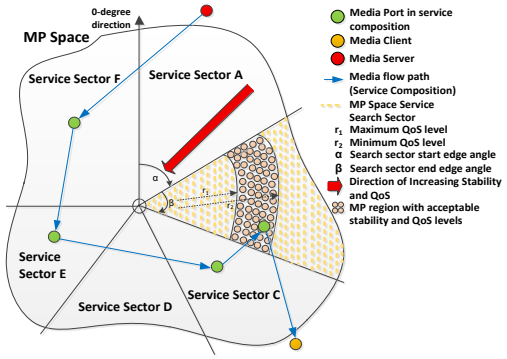

Figure 2. Potential MP search area in circle-based sectors.

We can utilize this overlay organization to search for nodes that meet the MC's service quality. This is done by finding a QoS upper and lower thresholds which when accumulated for each component service in the composition would maintain an acceptable level of QoS for the MC. In Fig. 2, r_2 represents the maximum distance away from the network's center where a node having the minimum acceptable level of stability and QoS can be chosen to perform the component service required for the composition. r_1 represents the distance away from the center of the network where nodes with the maximum level of stability and QoS exist. These nodes meet the minimum QoS requirements of the MC's service composition. Including nodes in the region with a radius less than r_1 is possible; however, it would mean a lower level of resource utilization for the chosen node. From the above, we can conclude that the overlay's geographical area searchable for a possible MP in a composition is given by the following equation:

$$SearchArea = \frac{\pi\beta}{360}(r_2^2 - r_1^2) \qquad (2)$$

Where β represent the component service's sector angle, r_2 represents the area's border with lowest acceptable level of QoS, and r_1 represents the area's border with highest expected QoS level. Any node receiving a query can then be added to a composition if following conditions from Eqn. 2 can be met:

$$(\alpha \le \theta \le (\alpha + \beta)) \wedge (D_{Low} \le D_{MP} \le D_{High}) \qquad (3)$$

Where Θ represents the MP's angle relative to the absolute 0° angle known by all nodes and D_{Low} represents the distance of any point on the arch r_1 distance away from the center of the MP space given by the following formula:

$$D_{Low} = \sqrt{(x_l - x_c)^2 + (y_l - y_c)^2} \quad (4)$$

Parameters:
NodeState = {Sleeping, Waiting, Processing, Active}
QoS_{Req} = Quality required by MC; QoS_{Curr} = current QoS level of the composition path; Q = Query description of service requested by MC.
MS (Initiator)
NodeState = Processing;
Use MS_{Out} and MC_{In} to evaluation possible MP types T = {T_1, T_2, …, T_n}; ***DO** (MP Search);*
for each T_i bounded by (α_j, β_j, r_{1j}, r_{2j}) → R=(α_j, β_j, r_{1j}, r_{2j});{
Send (Q, MS, MC, QoS_{Req}, QoS_{Curr}, R, Composition List (*Empty*)) } **end for**
NodeState = Waiting;
MP (Evaluation) Performed by MP receiving query
NodeState = Processing;
if (MP within R){ **if** (MP can process Q){ **Process**(Q);
Use MP_{Out} and MC to **evaluate** next-hop MP types T = {T_1, T_2, …, T_n}; ***DO** (MP_Search);* } **end if**
} **else** { evaluate next-hop MP types T = {T_1, T_2, …, T_n}; ***Do** (Find_Alternatives*); } **end if**
***DO** (Find_Siblings);*
procedure MP_Search
for (T = T_1 → T = T_n){
Evaluate possible search sectors Sect = { (α_1, β_1), (α_2, β_2), …, (α_n, β_n)}
for (i=1 → i=n){ Determine search radius of sector based on QoS_{Req} for each sector;
Given (α_i, β_i) **find** (r_{1i}, r_{2i});
for each search region bounded by (α_i, β_i, r_{1i}, r_{2i}){ **Send**(Q, MS, MC, QoS_{Req}, QoS_{Curr}, Composition List) } **end for** } **end for** } **end for**
procedure Find_Sibling
$MP_{inSibling}$=MP_{in}; $MP_{outSibling}$=MP_{out}; (α_x, β_x) = (α_{MP}, β_{MP}); Sibling must be within same sector
(r_{1x}, r_{2x}) = (r_{MP1}, r_{MP2}); Sibling must have same QoS range
Remove MP from current Composition List;
for region bounded by (α_x, β_x, r_x, r_x){ **Send** (Q, MS, MC, QoS_{Req}, QoS_{Curr}, CompositionList)} **end for**
procedure Find_Alternatives
MP_{inAlt}=MP_{in}; MP_{out} → MC_{in}
Do (*MP_Search*)
procedure Routing (Q, MS, MC, QoSReq, QoScurr, R, CompositionList)
If (my_type="minor") **then** { **If** can't process Q **then**
Send (Q, MS, MC, QoS_{Req}, QoS_{Curr}, R, CompositionList) to 'parent'} **end if** }
else if (my_type="major_link" || "major") **then** {
If (can't process Q & Q came from Chord neighbor) {
if (MP within R & QoS_{Curr}>QoS_{Req}) **then** {
If (one of 'minor' nodes meet requirements) **then** { send Q}
else {send to attached major_link node if any} **end if**
else if (MP within R & QoS_{Curr}<QoS_{Req}) { **Send** to neighbor with $QoS_{neighbor}$ > QoS_{Curr} }
else If (MP !Within R & there exist a neighbor with acceptable R) {
Send to neighbor such that ($R_{neighbor}$-R) < (R_{Curr}-R)}
else { If (my_type="major") { **Send** Q to 'major' parent node} **end if**} **end if**} **end if**

Algorithm1. Path formation.

Where x_l and y_l are the distances on the x- and y-axis of any point on the lower arch, and where xc and yc are the (0,0) point representing the MP space's center. The same can be applied to D_{High} by replacing x_l and y_l with x_h and y_h respectively. Using the above approach, we can rapidly decrease the search time for component services to form our composition and reduce the number of MPs where queries are sent. Using our hierarchical dynamic hybrid lower overlay, the service sector angle of search is reduced to an area guaranteed to contain the type of service required for a given composition. In addition, our use of node scores for node positioning in the overlay, the relative distance of required MPs can be utilized to further decrease number of potential nodes in a composition. Refer to algorithm 1 for the path formation process from the MS to the MC.

6 Performance Evaluation

In this section, we present the evaluations we did on the proposed lower overlay network structure using the C++-based OMNET++ [11] discrete event network simulator. This simulator permits the design of simulation models utilizing hierarchical architecture of a system module and its various sub-modules communicating via messages, gates, and links.

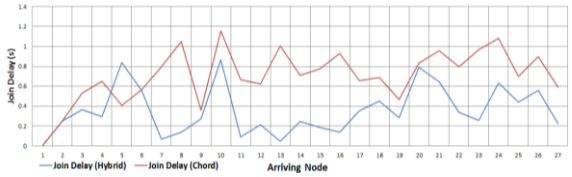

Figure 3. Join delay experienced by arriving nodes.

An open-source overlay and P2P network simulator OverSim [12] for OMNET++ was used to model Chord overlay networks and we extended the simulator to perform our proposed lower-overlay hybrid network simulation for the MP space. In our tests, nodes were placed in a 2-dimensional Euclidean space where delay between any two nodes is assumed proportional to the distance between them. In addition, a lifetime based churn model using Exponential distribution was used to model arrival and departure of nodes in the network. A sample application program of simple request-response type messages was used utilizing a full-recursive KBR protocol. KBR encapsulates messages and forwards them to the next hop according to the local routing table of a node closest to the node with the destination key. The message is then forwarded recursively back to originator.

In the simulation, control messages' packet sizes depended on the type of message exchanged, given the base message length of 32 bytes for join request messages. Each node was permitted a maximum of two join retries with an interval of 10 seconds before a join is determined as a failure. To maintain updated KBR. application test messages were set to a size of 100 bytes. Messages were generated with an interval of 60 seconds and a failure latency of 10 seconds. Chord stabilization messages were set with the following intervals: Chord finger reconfiguration with 120 seconds, predecessor update with 5 seconds, and stabilization with 20 seconds. Implementing the proposed hybrid overlay required modification to the routing protocol used by each node, and introducing new messages for *major* and *minor node* link stabilization and update, as well as link reconfiguration to form new low-level Chord structures and/or *minor nodes*.

In the first scenario, two simulation tests determined the average delay experienced by each node to join the Chord network compared with the hybrid overlay. An initially empty network was used with 10 sec. node inter-arrival rate. Results shown in Fig. 3 illustrate a visible difference in delay

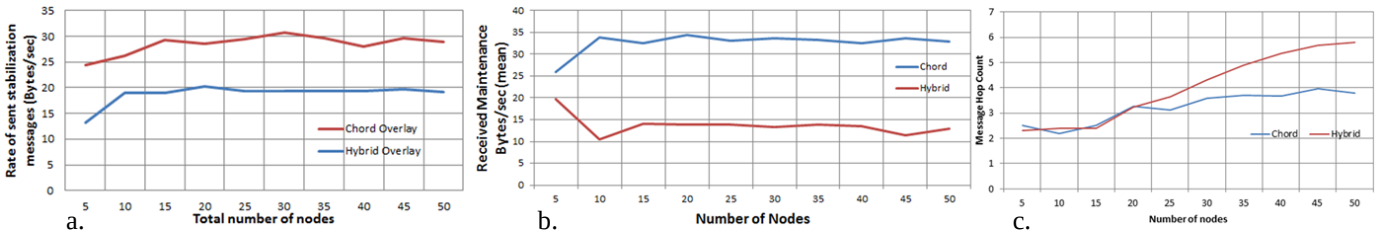

Figure 4. a. Mean rate of stabilization messages, b. Maintenance bytes received, c. Message hop count

experienced by nodes in the two networks. Joining a Chord structure with *N* nodes and *K* keys requires *O(K/N)* keys to change hands and may result in some delay before a join is completed. The same applies in the hybrid overlay, however, the presence of the leaf-like unstructured extension in the proposed overlay results in a significant decrease in the time required for a join since only the major node is contacted to complete the join process. Fig. 4a presents the comparison of the mean rate stabilization messages sent between nodes in the two networks. In Chord, with increasing number of nodes, the number of messages needed to maintain links to the successor, predecessor, and finger links increases until it plateaus to approximately 30 Bytes/sec. With the same number of nodes, the same stabilization messages are exchanged in the hybrid overlay. However, reduced number of nodes, present within each Chord structure and the distribution of other nodes in the leaf positions indicates a decrease in the number of stabilization messages sent. Even when considering the fact that our hybrid overlay introduces two stabilization messages, *Parent_Live_Stabilization* and *Leaf_Live_Stabilization* exchanged between *minor*- and their controlling *major- nodes* respectively, the mean number of stabilization messages sent by nodes in the overlay remains well below Chord networks.

Fig. 4b illustrates the expected high rate of maintenance messages received by each Chord node to fix broken links, while the hybrid structure illustrates a reduction of approximately 62% in rate maintenance messages arrival. Given the fact that *minor nodes* only receive messages from their *major nodes*, the low rate associated with hybrid overlay is understandable. One tradeoff noticed with the hybrid overlay is the number of hops a message must travel to reach its destination. Fig. 4c illustrates presence of a slight increase in the number of hops over that of the Chord network. This is expected, because with use of hybrid network, the worst case scenario exists when the source and destination are *minor nodes* at opposite ends of the overlay in a lower-level Chord structure. Such a limitation does not exist in a Chord network; however the benefits illustrated above of the hybrid overlay outweigh this slight increase in hop count.

7 Conclusion and Future Work

In this paper, we presented a dynamic hybrid service overlay network for MP distribution in mobile networks. We illustrated the dynamic characteristics of the overlay taking node scores based on stability, services provided, and QoS into consideration and service search method using a circular MP space for fast and accurate service compositions. Simulation tests have shown significant improvements of join delays, improvements in the mean rate of stabilization messages, and the mean rate of maintenance messages sent in our hybrid overlay in comparison with Chord networks. We are currently extending our study to provide a fast, context-aware, healable, and reconfigurable service composition method based on bounded approximations of fuzzy sets over our service hybrid overlay.

8 References

[1] E. Asmare, S. Schmid, M. Brunner, "Setup and Maintenance of Overlay Networks for Multimedia Services in Mobile Environments," in Proceedings of International Federation for Information, pp. 82-95, 2005.

[2] Z. Duan, Z. Zhang, Y.T. Hou, "Service overlay networks: SLAs, QoS, and bandwidth provisioning," in IEEE/ACM Transactions on Networking, vol. 11, issue 6, pp. 870-882, 2003.

[3] N. Niebert, A. Schider, H. Abramowicz, G. Malmgren, J. Sachs, U. Horn, C. Prehofer, H. Karl, "Ambient Networks: An Archictecture for Communication Networks Beyond 3G," IEEE Wireless Communications, vol.11, no.2, pp.14-22, 2004.

[4] S. Schmid, S. Herborn, J. Rey, "SMART: Intelligent Multimedia Routing and Adaptation based on Service Specific Overlay Networks," in Proceedings of Ubiquitous services and applications, EURESCOM'05, pp. 69-77. 2005.

[5] H. Guo, J. Liu, Z. Wang, "Frequency-Aware Indexing for Peer-to-Peer On-Demand Video Streaming," in IEEE International Conference on Communications, pp. 1-5, 2010.

[6] H. Han, J. He, C. Zuo, "A Hybrid P2P Overlay Network for Hight Efficient Search," in 2nd IEEE International Conference on Information and Financial Engineering, pp. 241-245, 2010.

[7] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," in the ACM Special Interest Group on Data Communication, pp.1-12, 2001.

[8] Z. Li, G. Xie, K. Hwang, Z. Li, "Churn-Resilient Protocol for Massive Data Dissimenation in P2P Networks," in IEEE Transactions on Parallel and Distributed Systems, pp. 1-8, 2009.

[9] I. Al-Oqily, A. Karmouch, "SORD: A Fault-Resilient Service Overlay for MediaPort Resource Discovery," in IEEE Transactions on Parallel and Distributed Systems, vol. 20, no. 8, pp. 1112-1125, 2009.

[10] A. Varga, "OMNeT++ User Manual, Version 4.1," [online report]http:// omnetpp.org/doc/omnetpp41/Manual.pdf

[11] D. Wang, C. Yeo, "Superchunk based Fast Search in P2P-VoD System," in IEEE Conference on Global Telecommunications, pp. 1-6, 2009.

[12] I. Baumgart, B. Heep, S. Krause, "OverSim: A Flexible Overlay Network Simulation Framework," in Proceedings of 10th IEEE Global Internet Symposium, p. 79-84, May 2007.