Performance Evaluation of Distributed Mobile Application Virtualization Services

Chung-Ping Hung* and Paul S. Min†
Department of Electrical and Systems Engineering, Washington University in St. Louis
One Brookings Drive, St. Louis, MO 63130, USA
Email: *chung23@wustl.edu †psm@wustl.edu

***Abstract*—In this paper, we first introduce how virtualization technologies can mitigate mobile application software publishing problems due to platform diversity and fragmentation. We propose a distributed server arrangement and the corresponding hand-off protocol to provide better user experience for application virtualization on mobile devices and evaluated the performance using the modified UMTS mobility models. We complete the establishment of quantitative relations between the performance improvement or impact and the infrastructure related parameters in the typical mobility model.**

Keywords-telecommunication; wireless networks; computer networks; information technology; UMTS mobility model

I. INTRODUCTION

As computing devices get smaller, lighter, and more portable, computing becomes more focused on mobile applications. We expect this trend to continue for years to come.

Deploying application software on mobile computing devices can be a challenge for several reasons. First of all, various mobile operating systems exist and none is expected to dominate and set the standards for the mobile computing in ways the Windows operating systems have done for the desktop computing. Making a software program to be compatible with different mobile operating systems requires extra cost and effort – for example, developing on multiple SDKs (Software Development Kits).

Although compatibility across application-platforms also exist on typical desktop computers, it is far more difficult for mobile computing devices because of the additional constraints such as limited compute cycles in the mobile devices. Unlike operating systems for desktop computers, mobile operating systems are highly customized per product and secured against unauthorized user access. Generally, ordinary end-users cannot upgrade or patch their mobile operating systems to address application-platform compatibility issues, as can be done for the desktop computers. It is thus hinged upon software developers to provide compatibility across mobile platforms.

Virtualization can address the compatibility in deploying mobile application software on various mobile platforms. Theoretically, we can either use application streaming to deploy application software over the Internet and run the application software on top of a preinstalled runtime environment, i.e., virtual machine, or run the application software on a managed server while each client device deals with user inputs, such as keystrokes, and outputs, such as display updates from the server [1], [2], [3]. We generally refer to the latter paradigm as the browser-based approach since web browsers provide an ideal framework for it.

Although technically plausible, deploying a virtual machine running on top of a mobile operating system provides an alternative way to distribute applications, which indeed violates the "Non-Compete" policy [4], [5] by marketplace operators[1]. Consequently, the browser-based approach becomes the only practical way to provide application virtualization on mobile computing devices.

The conventional solution of web-based application virtualization involves setting up a server or a group of servers at a co-location center (or data center) provided by an Internet service provider (ISP). From the co-location center, application virtualization services are provided through the Internet. This configuration typically incurs long response latency and significantly reduces the user experience since every input must travel through a series of routers and bridges to the co-location center and the corresponding response has to traverse backward through a similar route. Each node along the route introduces processing delay, queuing delay, and transmission delay.

To alleviate this issue, we propose an alternative configuration which partitions a service area into multiple smaller service areas with own server(s) [7]. The proposed configurations can significantly reduce delays since each server is closer to its user.

The proposed configuration, however, has to handle hand-off, i.e., mobile stations moving from one service area to another. We also propose a hand-off protocol offering seamless user experience in Section IV.

The proposed configuration comes with a price, such as introducing longer response latency during hand-offs. We use an analytical approach to evaluate the performance as a result of infrastructure arrangement [7].

We further set up a simulation environment based on the

[1]VMware's Mobile Virtualization Platform (MVP) [6], which implements this paradigm, is not available in Android Marketplace. To install MVP on an Android phone requires sideloading, and only Android platform leaves this loophole to install apps outside the marketplace, which is at the mercy of Google and wireless service providers. In fact, some wireless providers do block sideloading on some Android phones. Furthermore, among the major mobile device players, only Android is supported by MVP. Therefore, even VMware starts their own app store for MVP, it does not help cross-platform software deployment anyway.

UMTS urban pedestrian model and vehicular mobility model, and use the empirical approach to establish the correlations between the performance and the size of local service areas [8], [9].

II. Related Work

There are several papers proposed to optimize service migration though for different applications. Bienkowski et al. proposed competitive analysis for service migration in optimizing the server allocation in VNets in [10]. Arora et al. proposed some strategies for flexible server allocation in [11] following the previous work [10]. Although these works were not specifically for mobile application virtualization, they provide a precious insight on the performance evaluation for dynamic service allocation considering both user experience and operational cost. However, the analytical approach used in these works is topological and does not focus on the user mobility and interaction models. Furthermore, the authors of [10] and [11] allow services being temporarily interrupted during migrations, which is not feasible for application virtualization services. In the proposed configuration, application services are available to users with reduced performance during hand-offs.

III. Distributed Application Virtualization Service Configuration

Running application software on a remote server is conceptually similar to the usage model of time-sharing mainframe computers in the 1960s [12]. Although the communication bandwidth between terminals and mainframe servers at that time was low by the recent standards, it did not affect the user experience thanks to the text-only display and short traverse distance. However, in recent application virtualization technologies such as Virtual Desktop Infrastructure (VDI) proposed by VMware [13], much more complex and bloated content must be exchanged over much longer distances between clients and servers, especially for mobile users.

An infrastructure ready to offer mobile users application virtualization services includes base stations (BSs) covering the whole service area, a core network connecting base stations and servers together, and a server hosting the services. A command sent by a mobile station (MS) has to travel over the wireless channel to the BS, go through the backhaul network to the server, and then make some changes on the server. Should any update corresponding to the command be sent to the MS, the information has to travel all the way backward. In order to reduce the network delay generated by long transmission distances among the backhaul network, we deploy multiple servers among a wide area to serve their nearby MSs in the proposed configuration, instead of setting up a group of servers located at one data center serving all MSs.

In the proposed configuration, each server connects to several nearby BSs which form a local service group (LSG). The area covered by the BSs of the same LSG is defined as the local service area (LSA). Every BS belongs to one LSG in order to provide the service over the wireless network's coverage area. When a user demands a virtual application program, the server of the LSG, based on VDI [13] paradigm, starts a virtual machine (VM) dedicated to the user and launches the application software on top of it. The MS only handles inputs and outputs that interact with the VM at the server.

As long as the MS stays in the same LSA, the user can enjoy using application software with low response latency. If the MS moves from the original LSA to a nearby one, a hand-off at the VM level, which transfers the runtime environment to the server of the next LSG, is triggered. The detail of the hand-off protocol will be proposed in the next section.

IV. Hand-off Protocol

The purpose of the proposed hand-off protocol is to transfer minimum information required to recreate the runtime environment on a remote server, i.e., the snapshot, without interrupting the service. No matter how small the snapshot is, it still takes a period of time before the next server receives the complete snapshot and is ready to take over the service. In order to provide a seamless user experience during this period, the next server has to record all inputs from the MS, relay all inputs to the previous server, and relay all output from the previous server to the MS, until the runtime environment resumes locally. The proposed hand-off protocol is described as below:

1) When an MS moves from Server A's to Server B's LSA and sends an input command, Server B notices a newcomer within its LSA.
2) Server B broadcasts the newcomer's identification to all geographically nearby servers.
3) Server A, which hosts the MS's runtime environment, i.e., its VM server, responds Server B's inquiry. Now Server B knows the newcomer's VM server is Server A.
4) Server B records and relays the user's input commands to Server A, signals Server A to transfer the runtime environment, and relays display updates from Server A to the newcomer.
5) Once Server A is signaled to transfer the runtime environment, it takes a snapshot.
6) Besides continually responding to the input commands relayed from Server B as the MS is still in its LSA, Server A also sends the snapshot to Server B in the background.
7) Once Server B receives the complete snapshot and recreates the runtime environment from the snapshot and base data, it internally feeds the input queue, which was recorded during the transition period, to the runtime environment. Therefore, the runtime environment state on Server B is synchronous with that on Server A after the snapshot was transferred.
8) Server A completely stops serving the MS, the MS's VM server is now Server B instead.

The timeline of the proposed hand-off protocol is illustrated in Figure 1.

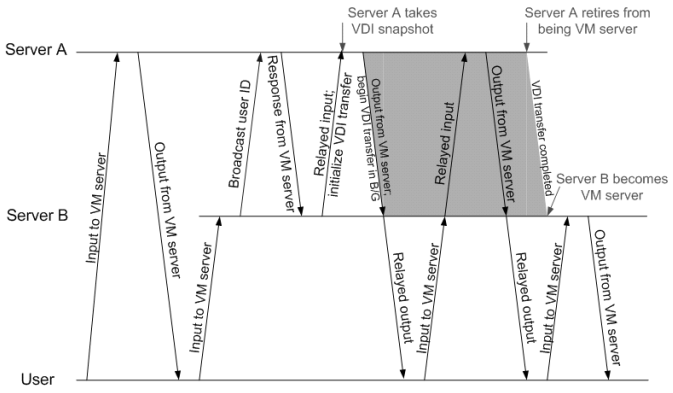

Fig. 1. Protocol timeline for a mobile station moving from Server A to Server B.

If the MS turns around and reenters Server A's LSA before the hand-off is completed, Server A can preempt the snapshot transmission and resume serving the MS as if the hand-off never happened. Since Server B relays all inputs to Server A while the MS is absent from Server A's LSA, aborting the hand-off procedure would not generate any glitch noticed by the user. This hand-off abortion mechanism can prevent unnecessary data transmission from moving VM servers back and forth if an MS were moving around the edge of an LSA.

On the other hand, if the MS moves to Server C's LSA before the hand-off is completed, Server C initializes another hand-off procedure with Server B. In addition to the snapshot, Server B has to transfer the input record before Server C joins the hand-off chain. We allow pipelining transmission to reduce hand-off periods and shorten subsequent hand-off chains in this scenario.

V. Performance Evaluation Using Free Particle Mobility Model

We define the response time as the average time interval between a user sends an input and gets an expected output update. The proposed server configuration is meant to improve the response time by reducing propagation delay along the communication route between each base station to the server which is hosting the service. Factors other than the propagation delay, such as wireless communication technologies and computational capabilities provided by servers, would affect the user experience and the quality of our service. Most of them, however, either affect both configurations equally, or can be overcome with reasonable cost.

The proposed configuration reduces the propagation delay and thus provides more responsive user experiences when uses are standing still. When a hand-off occurs, however, the user may experience longer response time waiting for the information to be exchanged between two servers before the runtime environment is successfully taken over by the new server. The smaller each local service area is, the higher occurrence probability of hand-offs the user may experience. Therefore, we have to quantitatively estimate and compare the propagation delays of the conventional and the proposed server configurations.

The precise propagation delay analysis depends on a wireless service provider's core network topology and its users' moving pattern record. Instead of acquiring those field data, we focus on the intrinsic properties of the two configurations. There are two approaches to estimate the average response time due to propagation delay; one assumes continuous service areas, the other is based on the optimal arrangement of base stations. The details are presented in the following subsections.

A. Continuous Service Area Approach

In this approach, we simplify the communication model between mobile stations and servers. Here are our assumptions:

1) The whole service area can be covered by a single server, or proximately by multiple servers, each having a regular hexagon shaped service area seamlessly tiled together as a service array.
2) A mobile station can directly communicate with the server everywhere in its (local) service area.
3) Users are uniformly distributed geographically in the beginning. Users can either move a certain distance in any direction, or stay at the same location for a while.
4) The propagation delay of each link is proportional to its length.
5) Each server's allocation is geographically optimized, that is, each server is located at the center of its (local) service area to reduce the average propagation delay.

The traverse time in our case is defined by:

$$T_{traverse} = 2 \cdot (1 - P_{HO}) \cdot (T_r + T_l) + 2 \cdot P_{HO} \cdot T_{HO} \quad (1)$$

where P_{HO} is the probability of transactions which either trigger hand-offs or occur during each handoff, T_r is the radio propagation delay, T_l is the line propagation delay, and T_{HO} is the prolonged traverse time during each hand-off according to the proposed protocol. To simplify the problem, we only compare the following three configurations covering the same amount of area:

- A A single server covering a regular hexagon service area of edge length L.
- B 7 servers, each covering a regular hexagon service area of edge length $\frac{L}{\sqrt{7}}$, as shown in Figure 2.
- C 12 servers, each covering a regular hexagon service area of edge length $\frac{L}{\sqrt{12}}$, as shown in Figure 3.

1) Average Transmission Distance: We can calculate the distance from an arbitrary point within each hexagon-shaped service area to the center of the area, where the optimal server is. Since we assume that our users' locations are uniformly distributed geographically in our service area, we can estimate the average transmission distance for each user in terms of edge length of the service area. Due to the symmetry of hexagons, the average distance from an arbitrary point within a hexagon to its center is equivalent to the average distance from an arbitrary point within a 30-60-90 triangle to the 30-degree vertex as shown in Figure 4.

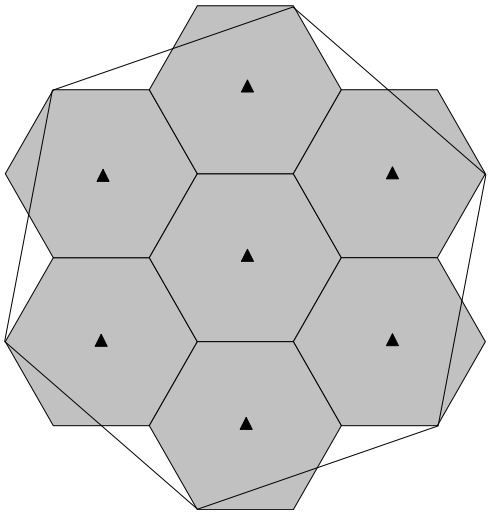

Fig. 2. Service area of 7-server configuration compares with of single-server one.

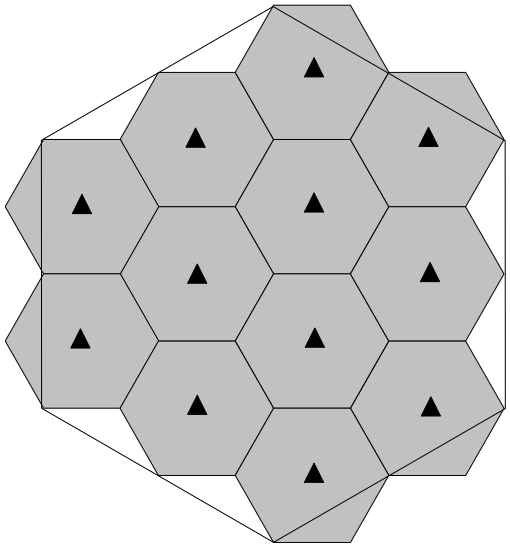

Fig. 3. Service area of 12-server configuration compares with of single-server one.

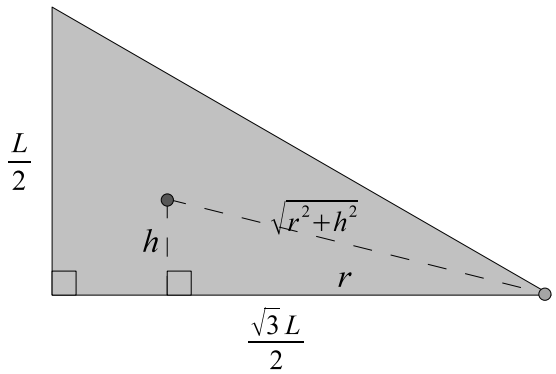

Fig. 4. 30-60-90 triangle as part of hexagon with edge length L, used to estimate average distance to the lower right vertex.

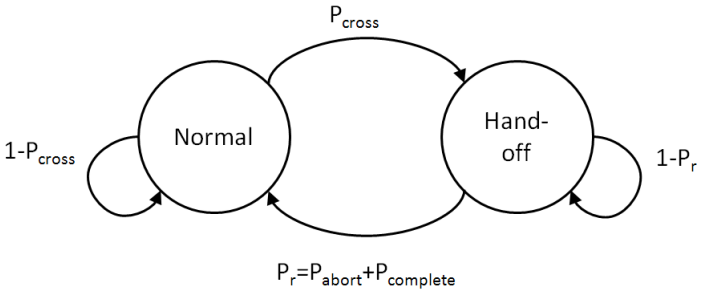

Fig. 5. The Markov chain of a moving MS's status.

By integrating $\sqrt{r^2+h^2}$ along h and r, as shown below:

$$\begin{aligned}
&\int_0^{\frac{\sqrt{3}L}{2}} \int_0^{\frac{r}{\sqrt{3}}} \left\{\sqrt{r^2+h^2}\right\} dh dr \\
&= \int_0^{\frac{\sqrt{3}L}{2}} \left\{\frac{h}{2}\sqrt{r^2+h^2} + \frac{r^2}{2}\ln\left|h+\sqrt{r^2+h^2}\right|\right\}\Bigg|_0^{\frac{r}{\sqrt{3}}} dr \\
&= \int_0^{\frac{\sqrt{3}L}{2}} r^2 dr \cdot \left\{\frac{1}{3} + \frac{\ln(3)}{4}\right\} \\
&= \frac{\sqrt{3}L^3}{8} \cdot \left\{\frac{1}{3} + \frac{\ln(3)}{4}\right\} \qquad (2)
\end{aligned}$$

By averaging the result above by the whole triangle area, the average transmission distance for each user in terms of the edge length of the service area is:

$$\frac{\frac{\sqrt{3}L^3}{8} \cdot \left\{\frac{1}{3} + \frac{\ln(3)}{4}\right\}}{\frac{\sqrt{3}L^2}{8}} = \left\{\frac{1}{3} + \frac{\ln(3)}{4}\right\} \cdot L \approx 0.60799L \qquad (3)$$

2) Probability of Transactions Relevant to Hand-off: The transition between normal and hand-off mode of each moving MS can be represented by a simple two-state Markov chain as shown in Figure 5.

As we can see in Figure 5, a moving MS in the normal state gets into the hand-off state when it moves across the border of its current local service area with probability P_{cross}. On the other hand, a moving MS in the hand-off state can go back to the normal state either by completing the hand-off procedure with probability $P_{complete}$, or by returning to the previous local service area and preempting the hand-off procedure with probability P_{abort}. The summation of $P_{complete}$ and P_{abort} is P_r, which represents the total probability for a moving MS in the hand-off state to return to the normal state.

By steady-state analysis, we can derive the probability of transactions relevant to hand-offs, i.e., P_{HO}, as below:

$$\begin{aligned}
\begin{bmatrix}1-P_{HO} & P_{HO}\end{bmatrix}\begin{bmatrix}1-P_{cross} & P_{cross} \\ P_r & 1-P_r\end{bmatrix} \\
= \begin{bmatrix}1-P_{HO} & P_{HO}\end{bmatrix} \\
P_{HO} = \frac{P_{cross}}{P_{cross}+P_r} \leq \frac{P_{cross}}{P_{cross}+P_{complete}} \qquad (4)
\end{aligned}$$

As we can see, the two factors P_{cross} and P_r affect P_{HO}. Both factors depend on users' mobility and the dimension of the service areas. Assume the average moving speed of an MS

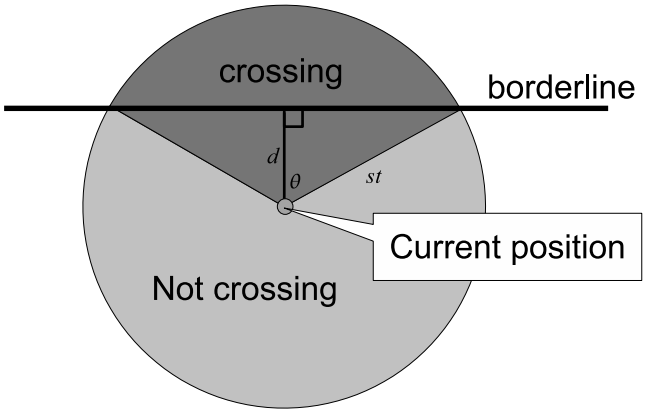

Fig. 6. For an MS close to the borderline who can freely choose it direction, the probability of crossing the borderline in the next time instance is $\frac{2\theta}{2\pi}$.

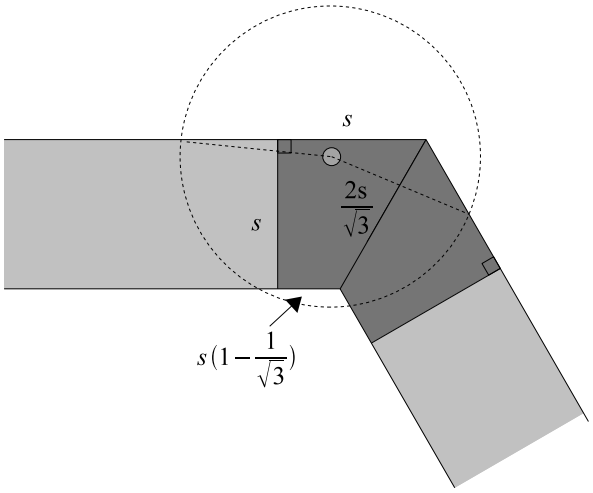

Fig. 7. Users at the singular area (dark area) have higher P_{cross}; the above equation can only apply in the normal areas (light areas).

is $\frac{s}{\Delta t}$. To make it possible to trigger a hand-off in the following time instance Δt, the MS has to be within s from the current service area's borderline. Furthermore, the probability of an MS satisfying this prerequisite actually crossing the borderline and thus triggers a hand-off depends on how close to the borderline it is as shown in Figure 6.

Therefore, the probability of an MS which is located d from the borderline with speed $\frac{s}{\Delta t}$ actually crossing the borderline in the next time instance Δt is given by:

$$P_{cross}(d,s) = \begin{cases} \frac{1}{\pi} \cdot \cos^{-1}\left(\frac{d}{s}\right) & 0 \le d \le s \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

However, since the service areas are hexagon-shaped, the borderline within the moving range is not always a straight line. As shown in Figure 7, the above equation does not apply to the MS located in the *singular area*, i.e., the area near vertices. It is so complex to estimate exact P_{cross} at singular area, such that we only calculate the range of P_{cross} instead.

We define $\hat{P}_{cross}(d,s)$ as the probability of an MS in the singular area crossing the borderline. Intuitively, the upper bound of $\hat{P}_{cross}(d,s)$ is $\frac{2}{3}$, in case of the MS starting at the corner, while the lower bound is $P_{cross}(d,s)$. The singular area would not be a problem in our estimation if L is relatively larger than s.

For each hexagon-shaped service area, the probability for an arbitrary MS crossing the borderline and triggering a hand-off is:

$$\bar{P}_{cross}(L,s,n) = \frac{2ns(L-2s)}{3\pi\sqrt{3}L^2} + \frac{2ns^2(2\sqrt{3}-1)\cdot\bar{\hat{P}}_{cross}}{9L^2} \quad (6)$$

where $\bar{\hat{P}}_{cross}$ is the average probability of an MS in the singular area crossing the borderline, and n is the number of edges which border another service area. The detail derivation will be presented in Appendix A.

Since $\bar{\hat{P}}_{cross} \le \frac{2}{3}$,

$$\begin{aligned}\bar{P}_{cross}(L,s,n) &\le \frac{2ns(L-2s)}{3\pi\sqrt{3}L^2} + \frac{2ns^2(2\sqrt{3}-1)\cdot\frac{2}{3}}{9L^2} \\ &= \frac{2ns\{3\sqrt{3}L + 2(2\sqrt{3}\pi - 2\pi - 3\sqrt{3})s\}}{27\pi L^2} \\ &= \frac{2\sqrt{3}n}{9\pi}\cdot\left(\frac{s}{L}\right) + \frac{4(2\sqrt{3}\pi - 2\pi - 3\sqrt{3})n}{27\pi}\cdot\left(\frac{s}{L}\right)^2 \end{aligned} \quad (7)$$

For $s \ll L$,

$$\bar{P}_{cross}(L,s,n) \approx \frac{2\sqrt{3}n}{9\pi}\cdot\left(\frac{s}{L}\right) \quad (8)$$

In the proposed design, the hand-off procedure takes a period of time while the MS can continue sending requests. The duration of a complete hand-off $T_{complete}$, as a result of the total amount of data which are transferred for each hand-off D_{sync}, the transmission bandwidth provided by the link between the two adjacent servers BW_s, and the transmission latency of the link T_{ls}, all affect the fraction of transactions relevant to hand-offs. The equation is given by:

$$T_{complete} = \frac{D_{sync}}{BW_s} + T_{ls} \quad (9)$$

Once a user triggers a hand-off, the subsequent requests within $T_{complete}$ are categorized as hand-off related transactions. In other word, there are at least $P_{complete} = \frac{1}{T_{complete}}$ of MSs in the hand-off status return to the normal status in average. The actual rate of leaving the hand-off status P_r should be substantially higher since some MSs preempt the hand-off. However, the hand-off abortion rate P_{abort} is very difficult to be derived with analytical approaches. Consequently, we take $P_{complete}$ as a reference of P_r first and discuss the relation between them later.

Now we can derive P_{HO} by the following equation:

$$\begin{aligned} P_{HO} &= \frac{\bar{P}_{cross}}{\bar{P}_{cross} + \frac{\alpha}{T_{complete}}} \\ &= \frac{\bar{P}_{cross}}{\bar{P}_{cross} + \frac{\alpha}{\frac{D_{sync}}{BW_s} + T_{ls}}} \\ &= \frac{\frac{2\sqrt{3}E(n)}{9\pi} \cdot \left(\frac{s}{L}\right)}{\frac{2\sqrt{3}E(n)}{9\pi} \cdot \left(\frac{s}{L}\right) + \frac{\alpha}{\frac{D_{sync}}{BW_s} + T_{ls}}} \\ &= \frac{2\sqrt{3}E(n)\left(\frac{s}{L}\right) \cdot \left\{\frac{D_{sync}}{BW_s} + T_{ls}\right\}}{2\sqrt{3}E(n)\left(\frac{s}{L}\right) \cdot \left\{\frac{D_{sync}}{BW_s} + T_{ls}\right\} + 9\pi\alpha} \end{aligned} \quad (10)$$

where $\alpha = \frac{P_r}{P_{complete}} > 1$ is the average hand-off duration, and $E(n)$ is the average number of edges which border another service area, which is 0, $\frac{24}{7}$, and 4 for Configuration A, B, and C, respectively.

We can roughly conclude that P_{HO} can be increased by higher MS mobility, a larger volume of the data required for the synchronization, and a longer transmission latency between the servers. On the other hand, it will be reduced by a wider service area, a higher bandwidth between the servers, and a higher rate of hand-off abortion. However, the transmission latency between the two adjacent servers is proportional to the service range. We will see how the service range affects the average response time in the following subsection.

3) Average Response Time Comparison of the Three Configurations: In Configuration A, there is only one server thus no hand-off mechanism. The average traverse time of Configuration A is quite straightforward:

$$T^A_{traverse} = 2 \cdot (T_r + T^A_l) \quad (11)$$

Now we have to consider hand-offs in Configuration B. Its average traverse time is:

$$\begin{aligned} & T^B_{traverse} \\ &= 2 \cdot (1 - P^B_{HO}) \cdot (T_r + T^B_l) + 2 \cdot P^B_{HO} \cdot T^B_{HO} \\ &= 2 \cdot (1 - P^B_{HO}) \cdot (T_r + \frac{T_l}{\sqrt{7}}) \\ &\quad + 2 \cdot P^B_{HO} \cdot (T_r + T_{lmax} + T_{ls}) \\ &= 2\left\{T_r + \frac{T_l}{\sqrt{7}} - \frac{P^B_{HO}T_l}{\sqrt{7}} + \frac{\left\{\frac{1}{2} + \frac{3\ln(3)}{4} + \sqrt{3}\right\}P^B_{HO}T_l}{\sqrt{7}\left(\frac{1}{3} + \frac{\ln(3)}{4}\right)}\right\} \\ &= 2\left\{T_r + \frac{T_l}{\sqrt{7}}\left\{1 + P^B_{HO}\left\{\frac{12\sqrt{3} + 2 + 6\ln(3)}{4 + 3\ln(3)}\right\}\right\}\right\} \end{aligned} \quad (12)$$

where T_{lmax} is the propagation delay between the MS and the new server during hand-offs, which is $\left\{\frac{1}{2\sqrt{3}} + \frac{3\ln(3)}{8\sqrt{3}}\right\}T_{ls}$, since we assume that the MSs are still located around the borderline at the time. The detail derivation, which is very similar to 2, will be presented in Appendix B.

Therefore, if we expect that Configuration B would outperform Configuration A, i.e., $T^B_{traverse} < T^A_{traverse}$, we can estimate the upper bound of P^B_{HO} as below:

$$1 + P^B_{HO}\left\{\frac{12\sqrt{3} + 2 + 6\ln(3)}{4 + 3\ln(3)}\right\} < \sqrt{7}$$

$$P^B_{HO} < \frac{(\sqrt{7} - 1)(4 + 3\ln(3))}{12\sqrt{3} + 2 + 6\ln(3)} \approx 0.4087356087 \quad (13)$$

This constrain is generally considered very slack.

Similarly, the average traverse time for Configuration C is:

$$\begin{aligned} & T^C_{traverse} \\ &= 2\left\{T_r + \frac{T_l}{\sqrt{12}}\left\{1 + P^B_{HO}\left\{\frac{12\sqrt{3} + 2 + 6\ln(3)}{4 + 3\ln(3)}\right\}\right\}\right\} \end{aligned} \quad (14)$$

And the upper bound of P^C_{HO} to outperform Configuration A is:

$$1 + P^C_{HO}\left\{\frac{12\sqrt{3} + 2 + 6\ln(3)}{4 + 3\ln(3)}\right\} < \sqrt{12}$$

$$P^C_{HO} < \frac{(\sqrt{12} - 1)(4 + 3\ln(3))}{12\sqrt{3} + 2 + 6\ln(3)} \approx 0.6119795056 \quad (15)$$

Furthermore, to outperform Configuration B given the same BW_s, the criteria is:

$$\begin{aligned} & \frac{T_l}{\sqrt{12}}\left\{1 + P^C_{HO}\left\{\frac{12\sqrt{3} + 2 + 6\ln(3)}{4 + 3\ln(3)}\right\}\right\} \\ &\quad < \frac{T_l}{\sqrt{7}}\left\{1 + P^B_{HO}\left\{\frac{12\sqrt{3} + 2 + 6\ln(3)}{4 + 3\ln(3)}\right\}\right\} \\ &\Rightarrow \frac{8\sqrt{21}\left(\frac{\sqrt{12}s}{L}\right) \cdot \left\{\frac{D_{sync}}{BW_s} + \sqrt{\frac{7}{12}}T^B_{ls}\right\}}{8\sqrt{3}\left(\frac{\sqrt{12}s}{L}\right) \cdot \left\{\frac{D_{sync}}{BW_s} + \sqrt{\frac{7}{12}}T^B_{ls}\right\} + 9\pi\alpha_C} \\ &\quad - \frac{\frac{96}{7}\left(\frac{\sqrt{7}s}{L}\right) \cdot \left\{\frac{D_{sync}}{BW_s} + T^B_{ls}\right\}}{\frac{16\sqrt{3}}{7}\left(\frac{\sqrt{7}s}{L}\right) \cdot \left\{\frac{D_{sync}}{BW_s} + T^B_{ls}\right\} + 3\pi\alpha_B} \\ &\quad < \frac{(\sqrt{12} - \sqrt{7})(4 + 3\ln(3))}{12\sqrt{3} + 2 + 6\ln(3)} \end{aligned} \quad (16)$$

The detail derivation will be presented in Appendix C.

The inequality above provides an accurate bound of the function of $\frac{s}{L}$, $\frac{D_{sync}}{BW_s}$, T^B_{ls}, α_B, and α_C. It is, however, too complex to help us to determine which configuration is better given a set of system parameters. Fortunately, we can discover the benefit brought by a more distributed infrastructure arrangement by simplify the inequality above based on sensible approximations. First of all, in most case T_{ls} is negligible comparing to $\frac{D_{sync}}{BW_s}$. Therefore, we can replace all $T_{complete}$ by $\frac{D_{sync}}{BW_s}$. Secondly, as $s \ll L$, P_{abort}'s in both configurations are approximately the same. In consequence, $\alpha_B \approx \alpha_C$. Therefore, we can further set $\alpha P_r = \beta P^B_{cross}$ where $\beta > 0$ to simplify the inequality.

Since

$$\hat{P}_{cross}(L, s) \approx \frac{2\sqrt{3}E(n)}{9\pi} \cdot \left(\frac{s}{L}\right) \quad (17)$$

Therefore,

$$\frac{P^C_{cross}}{P^B_{cross}} = \frac{\frac{2\sqrt{3}\cdot 4}{9\pi}\cdot\left(\frac{\sqrt{12}s}{L}\right)}{\frac{2\sqrt{3}\cdot\frac{24}{7}}{9\pi}\cdot\frac{\sqrt{7}s}{L}}$$

$$\Rightarrow P^C_{cross} = \sqrt{\frac{7}{3}}P^B_{cross} \quad (18)$$

Now we can represent P^B_{HO} and P^C_{HO} only in terms of P^B_{cross} and β:

$$P^B_{HO} = \frac{P^B_{cross}}{P^B_{cross}+\alpha P_r} = \frac{P^B_{cross}}{P^B_{cross}+\beta P^B_{cross}} = \frac{1}{1+\beta}$$

$$P^C_{HO} = \frac{P^C_{cross}}{P^C_{cross}+\alpha P_r} = \frac{\sqrt{\frac{7}{3}}P^C_{cross}}{\sqrt{\frac{7}{3}}P^C_{cross}+\beta P^B_{cross}} = \frac{\sqrt{7}}{\sqrt{7}+\sqrt{3}\beta} \quad (19)$$

And then rewrite the inequality in terms of β:

$$\sqrt{7}P^C_{HO} - \sqrt{12}P^B_{HO} < \frac{(\sqrt{12}-\sqrt{7})(4+3\ln(3))}{12\sqrt{3}+2+6\ln(3)}$$

$$\frac{7}{\sqrt{7}+\sqrt{3}\beta} - \frac{2\sqrt{3}}{1+\beta} < \frac{(\sqrt{12}-\sqrt{7})(4+3\ln(3))}{12\sqrt{3}+2+6\ln(3)}$$

$$\frac{(7-2\sqrt{21})+\beta}{\sqrt{7}+(\sqrt{7}+\sqrt{3})\beta+\sqrt{3}\beta^2} < \frac{(\sqrt{12}-\sqrt{7})(4+3\ln(3))}{12\sqrt{3}+2+6\ln(3)}$$

$$\Rightarrow 0.352\beta^2 - 0.110\beta + 2.703 > 0 \quad (20)$$

which is true for all β.

Therefore, we can conclude that if $s \ll L$, $\frac{D_{sync}}{BW_s} \gg T_{ls}$, and $\alpha_B \approx \alpha_C$, Configuration C can always outperform Configuration B in terms of traverse delay.

4) The Actual Rate of Leaving Hand-off State: To better understand P_r and its relation to $T_{complete}$, we wrote a simple simulation program to empirically measure the average time an MS stays in the hand-off status. The program simulates an MS originally located close to a borderline, whose distance to it is uniformly distributed from 0 to s. Before it cross the borderline and triggers a hand-off, it randomly choose a direction from $-\pi$ to π and step forward s, which ensures it either crosses, or approaches to, the borderline. Once it triggers a hand-off, it can randomly choose any direction to step forward until the predetermined $T_{complete}$ runs out or it moves back to the other side of the borderline. The time each MS stays in the hand-off status is gauged and averaged in the end of the program.

The average time MSs stay in the hand-off status given different $T_{complete}$ is shown in Figure 8.

The value of α, which varies in a similar curve as in Figure 8 is shown in Figure 9.

As we can see in Figure 8 and Figure 9, the actual rate of leaving hand-off state P_r, which is the inverse of actual average hand-off duration, only fluctuates slightly in response to $T_{complete}$. Therefore, we can assure that ignoring T_{ls} and subsequently assuming that $T_{complete}$'s are identical for both configurations are sensible.

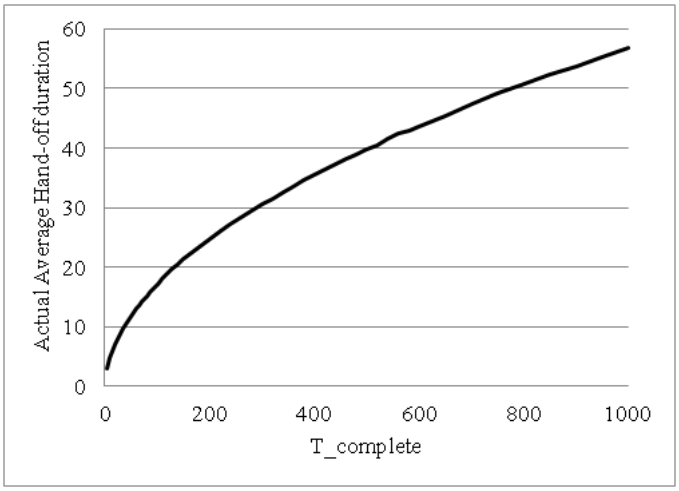

Fig. 8. The actual average hand-off duration.

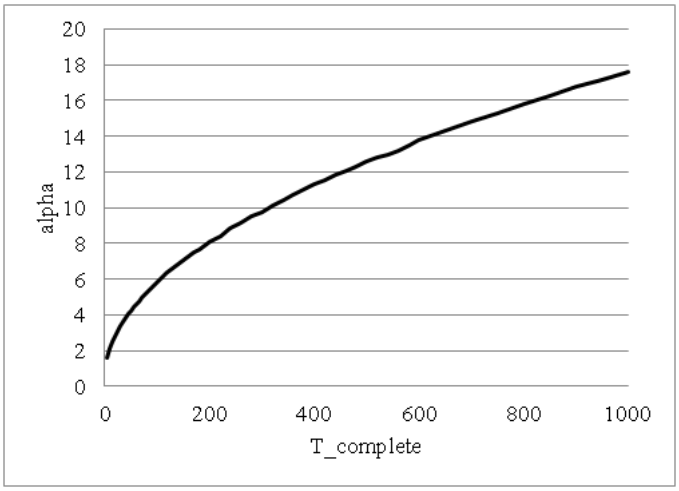

Fig. 9. The value of α given different $T_{complete}$.

B. Optimal Arranged Base Stations Approach

In this approach, the service area is covered by a group of base stations, each connected to a server. Unlike the continuous service area approach which assumes each service area is a perfect regular hexagon, in this model the service areas are shaped by overlapping disks, each covered by a base station with omni-directional antenna. Consequently, each (local) service area is similar to a regular hexagon but with some "ripples" around the edges, which make it very difficult to estimate the hand-off probability. We can, however, proximately estimate it in certain conditions. Here are the assumptions, which are slightly different from those of the other approach:

1) The whole service area is covered by minimum number of base stations with omni-directional antennae. In other

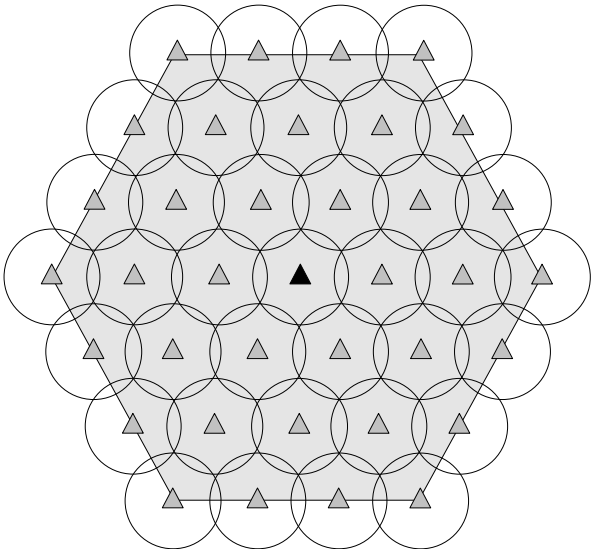

Fig. 10. Service area of single-server configuration with $m = 3$.

words, base stations are located at unit points of a two-dimensional Synergetics coordinates [14].

2) We can either connect all base stations to one server, or separate base stations into several groups and connect them to the server of each group. The optimal service area of each group is approximately a regular hexagon.
3) Users are uniformly distributed geographically in the beginning. Users can either move a certain distance in any direction, or stay at the same location for a while.
4) The propagation delay of each link is proportional to its length.
5) Each server's allocation is geographically optimized, that is, each server is located in the center of its (local) service area to reduce average propagation delay. The traverse time in our service is defined by (1) as well.

Again, we compare the following two configurations covering the same area.

A: $(3m^2 + 3m + 1)$ base stations are placed like a regular hexagon, where m is the number of the base stations' intervals along one of the hexagon's edges. Each interval is $\sqrt{3}R$ long, where R is the effective communication range of each base station. An example is illustrated in Figure 10.

B: 7 servers, each connected to $(3\lceil\frac{m}{3}\rceil^2 + 3\lceil\frac{m}{3}\rceil + 1)$ base stations as a local service area. The base stations in each local service area are placed like a regular hexagon with $\lceil\frac{m}{3}\rceil$ intervals along one of its edge, as shown in Figure 11.

1) Average Transmission Distance: The average transmission distance in this approach is the discrete version of the continuous service area's counterpart. However, it is very difficult to represent in terms of m, as shown below:

$$\frac{3r\sum_{t=0}^{m-1}\sum_{k=1}^{m-t}\sqrt{3(2k+t)^2+9t^2}}{3m^2+3m+1} \quad (21)$$

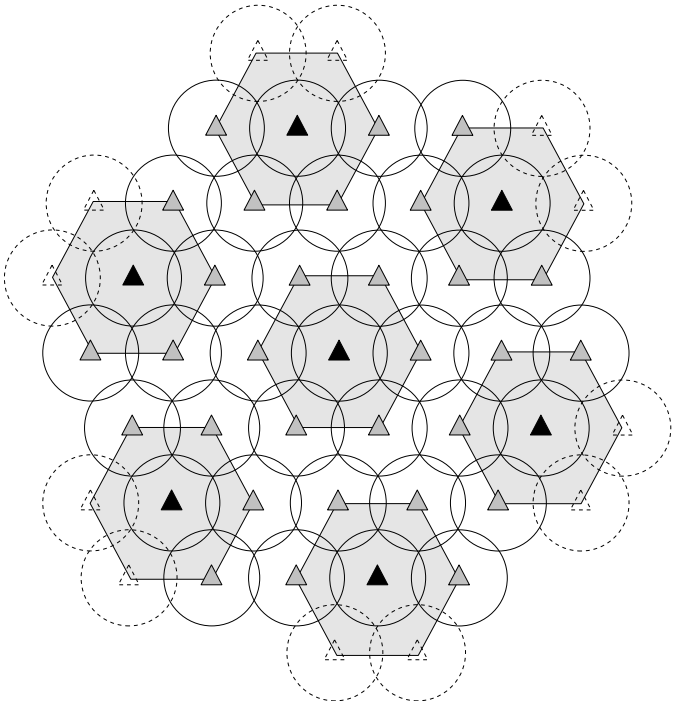

Fig. 11. Service areas of 7-server configuration, each with $m = 1$, covering the same amount of area.

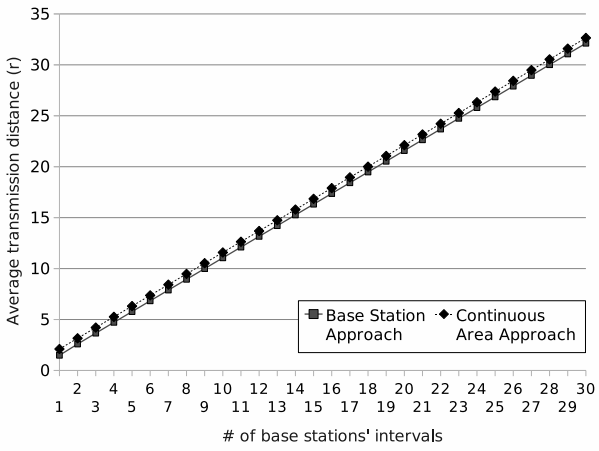

Fig. 12. Comparison of average transmission distances of different approaches covering approximately equal service area.

Fortunately, we find out that the average transmission distance in this approach is approximately linear and gets closer to its continuous counterpart as m increases according to the computer calculation, as shown in Figure 12.

In other words, we can estimate the average transmission distance by either (21), or the continuous counterpart (3) with comparable parameters. In the later sections, we will use the latter one to focus on the quantitative relationships between the parameters and the performance rather than the exact value.

2) Probability of Transactions Relevant to Hand-offs: Due to the irregular shape of each local service area, it is difficult to estimate the exact probability of an arbitrary user around the

border moving out of the service area by equations. However, if users' moving distances in each time instance are relatively short compared to a base station's effective communication range, the perimeter of each local service area at any point is near a straight line from a user's point of view.

Therefore, we can borrow the results from the continuous counterpart (6) to estimate the probability of a user crossing the borderline. The average probability of a user crossing the borderline along an arbitrary line which is perpendicular to the assumed straight borderline is:

$$\begin{aligned}\delta\bar{P}_{cross}(s) &= \int_0^s \left\{\frac{1}{\pi}\cos^{-1}\left(\frac{d}{s}\right)\right\}dd \\ &= \frac{s}{\pi}\int_0^1 \cos^{-1}(k)dk \\ &= \frac{s}{\pi}\left\{k\cos^{-1}(k) - \sqrt{1-k^2}\right\}\Big|_0^1 = \frac{s}{\pi}\end{aligned} \tag{22}$$

The perimeter of the service area has to be recalculated as $6(m-1)$ one-third arcs and 6 half circles of radius R:

$$6L_{edge} = 6(m-1)\cdot\left(\frac{2\pi R}{3}\right) + 6\cdot\left(\frac{2\pi R}{2}\right) = 2\pi R(2m+1) \tag{23}$$

For a local service area with n edges bordering another one, the length of borderline eligible to invoke hand-offs is:

$$nL_{edge} = \frac{n\pi R(2m+1)}{3} \tag{24}$$

And we recalculate the service area as well. The area is basically a hexagon with some "decorations" around the perimeter:

$$\begin{aligned}A &= \frac{3\sqrt{3}}{2}\left(\sqrt{3}mR + \frac{R}{\sqrt{3}}\right)^2 \\ &\quad + 6\left\{\frac{\pi R^2}{2} - \frac{R^2}{\sqrt{3}} + (m-1)\left(\frac{\pi R^2}{2} - \frac{\sqrt{3}R^2}{4}\right)\right\} \\ &= R^2\left\{\frac{9\sqrt{3}m^2}{2} + \left(2\pi + \frac{3\sqrt{3}}{2}\right)m + \pi\right\}\end{aligned} \tag{25}$$

By accumulating the $\delta\bar{P}_{cross}$ along the perimeter and averaging with total area, the probability of a user crosses the borderline for mobile stations located in the service area for $s \ll R$ is:

$$\begin{aligned}\bar{P}_{cross} &= \frac{n\pi R(2m+1)s}{3\pi R^2\left\{\frac{9\sqrt{3}m^2}{2} + \left(2\pi + \frac{3\sqrt{3}}{2}\right)m + \pi\right\}} \\ &= \frac{n(2m+1)s}{3R\left\{\frac{9\sqrt{3}m^2}{2} + \left(2\pi + \frac{3\sqrt{3}}{2}\right)m + \pi\right\}}\end{aligned} \tag{26}$$

Similar to the continuous counterpart, P_{HO} is given by the following equation:

$$\begin{aligned}P_{HO} &= \frac{P_{cross}}{P_{cross} + P_r} \\ &= \frac{\frac{E(n)(2m+1)s}{3R\left\{\frac{9\sqrt{3}m^2}{2} + \left(2\pi + \frac{3\sqrt{3}}{2}\right) + \pi\right\}}}{\frac{E(n)(2m+1)s}{3R\left\{\frac{9\sqrt{3}m^2}{2} + \left(2\pi + \frac{3\sqrt{3}}{2}\right) + \pi\right\}} + \frac{\alpha}{\frac{D_{sync}}{BW_s} + T_{ls}}}\end{aligned} \tag{27}$$

for $s \ll R$, where $E(n)$ is the average number of edges bordering another local service area as well, which is 0 and $\frac{24}{7}$ in Configuration A and B, respectively.

3) Average Response Time Comparison of the Two Configurations: The average response time of configuration A $T^A_{response}$ is still $2(T_r + T^A_l)$.The average traverse time of Configuration B is equal to its continuous counterpart (12) as well. If we expect that Configuration B would bring a shorter average response time over Configuration A, the upper bound of P^B_{HO} is unchanged:

$$P^B_{HO} < \frac{(\sqrt{7}-1)(4+3\ln(3))}{12\sqrt{3}+2+6\ln(3)} \approx 0.4087356087$$

Therefore, the constraints for m, s, R, $\frac{D_{sync}}{BW_s}$, α, T_{ls}, and T_u are represented in the equation below:

$$\frac{\frac{8(2m+1)s}{R\left\{\frac{9\sqrt{3}m^2}{2} + \left(2\pi + \frac{3\sqrt{3}}{2}\right) + \pi\right\}}}{\frac{8(2m+1)s}{R\left\{\frac{9\sqrt{3}m^2}{2} + \left(2\pi + \frac{3\sqrt{3}}{2}\right) + \pi\right\}} + \frac{7\alpha}{\frac{D_{sync}}{BW_s} + T_{ls}}} < 0.051091951 \tag{28}$$

C. Simulation Result

To verify the estimations of P_{cross}, we use the Monte Carlo method by running a simulation program which sets up base stations of given R at optimal locations, randomly puts a large number of mobile stations, moves them away from their original location a fixed distance in any direction, and measures the number of the mobile stations escaping from the service area.

To compare the errors of the two different approaches, we set two environments with short R and large m, and long R with small m, and adjustable s. In the former environment, we set $R = 0.25$, $m = 40$, s varies from 0.1 to 2.0 with 0.01 steps, and place 10^7 mobile stations. The P_{HO} derived by the estimators and measured in the simulation are compared in Figure 13.

As we can see, the continuous service area approach is a better estimator since the shape of the service area is very close to a perfect regular hexagon in this environment. Furthermore, we compare the error rate of both estimators and compare them in Figure 14.

We can see in this series of simulations, the optimal arranged base stations approach only works well with very low s. However, when we set $R = 2.0$ and $m = 5$ and run the same simulations, it becomes a different story as shown in Figure 15.

Since the base stations are far less dense than in the previous setting, the "ripples" around the service area get larger and distort the shape away from a perfect regular hexagon. As

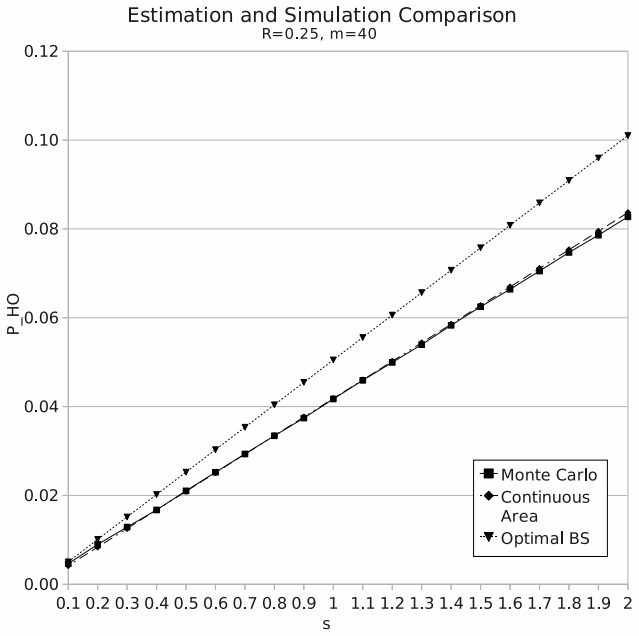

Fig. 13. Comparison of estimations and simulation result with $R = 0.25$ and $m = 40$.

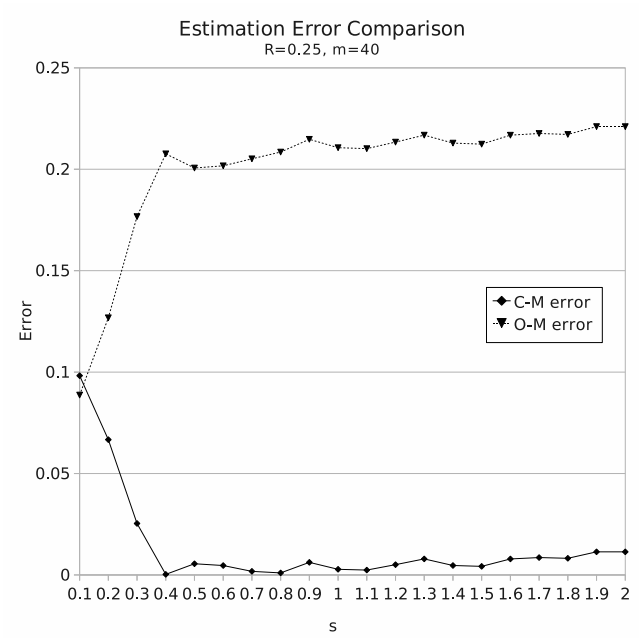

Fig. 14. Comparison of estimation errors with $R = 0.25$ and $m = 40$.

we can see in Figure 15, the optimal arranged base stations approach is a very accurate P_{HO} estimator for $s \leq 0.5$ ($s \leq \frac{R}{4}$), and the continuous service area approach gets more and more accurate P_{HO} in response to increasing user mobility.

By comparing the estimation errors of both approaches in Figure 16, we can see the accuracies of the two estimators significantly depend on user mobility.

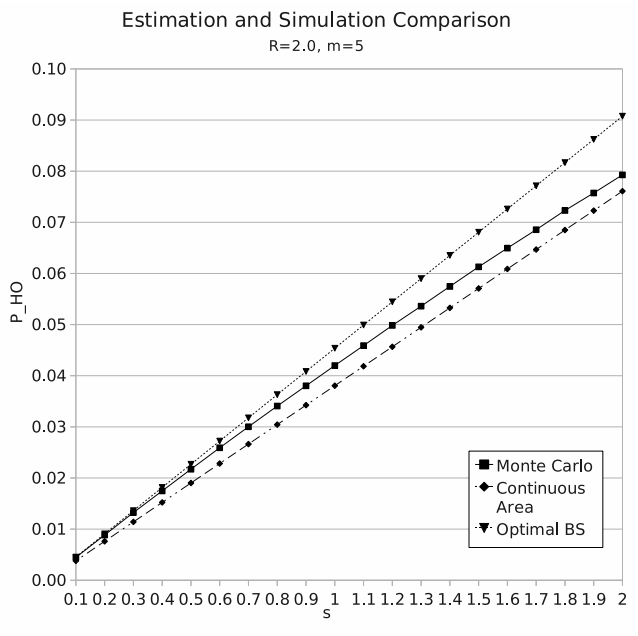

Fig. 15. Comparison of estimations and simulation result with $R = 2.0$ and $m = 5$.

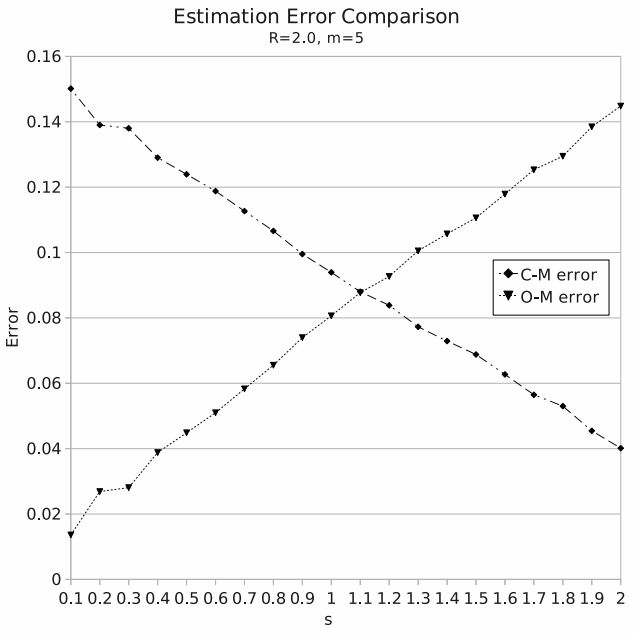

Fig. 16. Comparison of estimation errors with $R = 2.0$ and $m = 5$.

VI. Performance Evaluation Using the UMTS Urban Mobility Model

Although the free particle analysis in the previous section relates the overall performance to the MS's mobility and the infrastructure's geographical parameters, this mobility model is too arbitrary to preview the performance of the proposed configuration and hand-off protocol in real world. Since each MS moves without inertia and any intelligent intent, it is able to suddenly turn to an opposite direction in the free particle

mobility model. This moving characteristic only makes sense for an application virtualization service specific for a bunch of drunks wondering on a rural plain. In the proposed distributed service configuration, the free particle model does increase the chance an MS triggers and aborts a hand-off procedure in a short period. Therefore, we need to further evaluate the performance of the proposed configuration and hand-off protocol in more realistic mobility models.

Of course the most realistic usage model comes from the field statistics of a mobile phone carrier. The data is extremely difficult to be obtained for several reasons. For instance, carriers may record the users' moving pattern along with other behaviors and store them in a huge database in general. In most case, they do not do any data mining or organization except for their internal research projects. If an outsider requests a data set about user mobility from a mobile phone carrier, they do not know where the data is even if they are willing to help. Furthermore, those records may involve sensitive user privacy. Mobile phone carriers would reluctant to allow any outsider to get access to the databases to prevent from potential legal issues.

Fortunately, European Telecommunication Standards Institute (ETSI) published a document [15] which described three test environments and user mobility models, which are Indoor Office, Outdoor to Indoor and Pedestrian, and Vehicular ones, as common benchmarks to evaluate potential wireless technologies to develop Universal Mobile Telecommunication System (UMTS). Although the reality of the models is never explicitly justified and Jugl and Boche [16] have extended the mobility model to improve the reality, the original UMTS models still provide a fair reference for mobility related performance evaluation. If more realistic mobility models are available, we can replace the UMTS ones and obtain more accurate configuration parameters.

In this section, we set up a simulation environment referring to the UMTS's Outdoor to Indoor and Pedestrian mobility model, also known as the UMTS urban mobility model, and use the empirical approach to establish the correlations between the performance and the size of each local service area and the capabilities of the network infrastructure. With our proposed modification, we enable the simulation to run for an indefinite period of time without presuming any boundary condition.

A. UMTS Urban Mobility Model

As shown in Figure 17, the UMTS Outdoor to Indoor and Pedestrian test environment is basically a Manhattan-like street structure where MSs move along 30 meters wide streets and are only allowed to change directions with half chance at the intersections, which are 200 meters apart. Each MS's moving speed can be updated every 5 meters with 20% chance, and the new speed is generated by a truncated Gaussian distribution whose mean equals 3 km/h, standard deviation equals 0.3 km/h, and minimum speed equals 0 km/h. All MSs are initially uniformly distributed on the Manhattan-like streets.

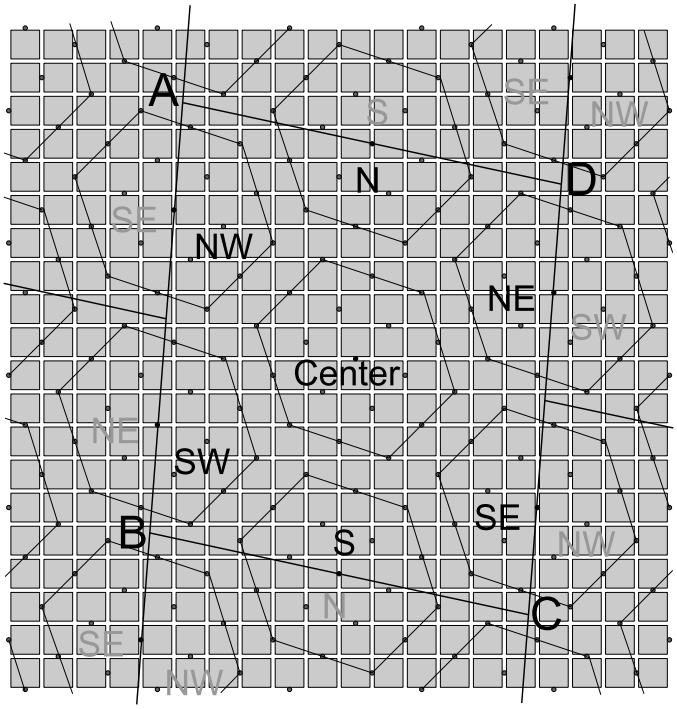

Fig. 17. UMTS outdoor to Indoor and Pedestrian test environment and LSA arrangement.

The UMTS document, however, does not explicitly specify where an MS turns within the intersection area. Therefore, we make a reasonable assumption to overcome the ambiguity. If an MS is supposed to turn in an intersection, it has six points, which are 5 meters apart along the crosswalk, to change its direction before reaching the other side. We assume an MS picks one out of the six points with equal chances as its turning point, which keeps MSs uniformly distributed on the streets rather than be concentrated on a certain part of the streets over time.

The BSs in the UMTS Outdoor to Indoor and Pedestrian test environment are located at the dark grey dots in Figure 17. Although the placement of the BSs is not optimal, it is not far from that. Considering an actual city could be preoccupied by tall private buildings on each block, deploying BSs along the streets makes sense both technically and politically.

One of the shortcomings of the UMTS mobility model is the bounded test area which generates ambiguities on setting boundary conditions. We consequently add some special traffic rules, known as *portals*, to eliminate the boundary discontinuities and allow the interaction among LSAs to be simulated and observed for indefinite period of time. These portals will be described in the next subsection.

B. Möbius City

What interests us is the geographical relation between the service facilities and the MSs' moving space. Once we group the BSs in Figure 17 to form hexagon-shaped LSAs that optimize in both coverage and average transmission distance by deploying servers at the centers, we can find a regular

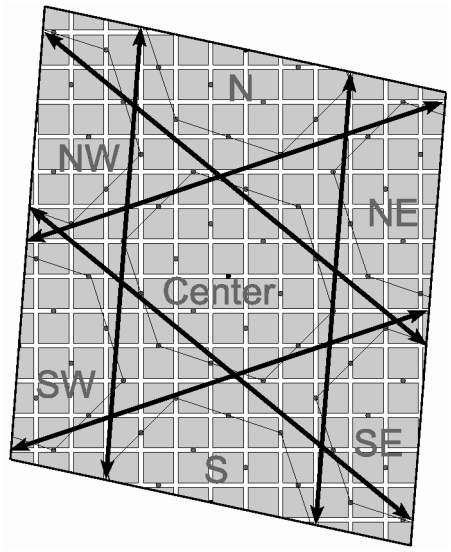

Fig. 18. Möbius City map with teleporting directions.

repetitive pattern of streets and service groups, which depends on N, the number of the BSs per LSA's edge. If we align the origin to a BS, the parallelogram ABCD surrounded by four straight lines, which are:

1) $(3N-1)x+(9N+5)y = 920(6N^2+6N+2)$ on the north,
2) $(3N-1)x+(9N+5)y = -920(6N^2+6N+2)$ on the south,
3) $(5N+3)x-(N-1)y = -920(3N^2+3N+1)$ on the west,
4) and $(5N+3)x-(N-1)y = 920(3N^2+3N+1)$ on the east,

can be regarded as the element of the repetitive pattern and represent sufficient geographical information we need. We can, therefore, crop out parallelogram ABCD in Figure 17 as our new test area, where we call *Möbius City* as shown in Figure 18, to represent every identical piece comprises the indefinite large test area.

Möbius City only has four LSGs. The center one is the only complete LSA. The north half (N) and the south half (S), the northwest half (NW) and the southeast half (SE), and the northeast half (NE) and the southwest half (SW), comprise the three other LSAs. The latter three LSAs' allocation emulates six complete LSAs around the center one in the original test area. Since we are only interested in when, where, and how frequently an MS moves from one LSA to another rather than specifically identifying which one it moves from and to, assigning only four LSGs is sufficient for our work.

Möbius City is comprised by the area cropped from the original street structure and portals at the boundaries. Just like moving through the tunnels in Pac-Man's maze, whenever an MS moving among the streets reaches a boundary and is about to escape from Möbius City, the portal teleports it to a proper location at the opposite side and reenter Möbius City. The rules of the portals are:

1) For MSs about crossing north boundary, teleport them to $(-230(N-1), -230(5N+3))$ from their current locations.
2) For MSs about crossing south boundary, teleport them to $(230(N-1), 230(5N+3))$ from their current locations.
3) For MSs about crossing west boundary and their current locations satisfy $3(N-1)x+(9N+5)y > 0$, teleport them to $(230(4N+3), -230(4N+1))$ from their current locations.
4) For MSs about crossing west boundary and their current locations satisfy $3(N-1)x+(9N+5)y \leq 0$, teleport them to $(230(5N+2), 230(N+2))$ from their current locations.
5) For MSs about crossing east boundary, and their current locations satisfy $3(N-1)x+(9N+5)y > 0$, teleport them to $(-230(5N+2), -230(N+2))$ from their current locations.
6) For MSs about crossing east boundary, and their current locations satisfy $3(N-1)x+(9N+5)y \leq 0$, teleport them to $(-230(4N+3), 230(4N+1))$ from their current locations.

The teleport directions are shown in Figure 18 as well.

An MS moving through a portal doesn't encounter any discontinuity except its coordinates: its direction and speed are the same, it associates with the same LSG, and the geographical parameters relative to the service group's facilities remain. Thus, everything interests us is equivalent as the MS moving into an adjacent parallelogram area in an indefinite large test area.

C. Configuration of Backhaul Network

Although connecting every BS to the corresponding server through a line-of-sight and high-speed direct link offers the lowest transmission latency, constructing such a backhaul network is impractically expensive. Therefore, we assume each BS only has direct connections to its six neighboring BSs to form a mesh network as the core network. In mesh-styled backhaul network, network latency between a BS and the server depends on the number of nodes along the shortest path, the total length of the path, and the relay latency per node. The former two factors are related to the coordinates of the BS and the server, which will be simulated as well.

D. Traverse Delay

We define the response time as the average time interval from a user sends an input till the expected output update is received. The proposed server configuration is meant to improve the response time by reducing traverse delay along the communication route between each BS to the server which is hosting the service. Factors other than the traverse delay, such as computational capabilities provided by servers, would affect the user experience and the quality of our service. Most

of them, however, either affect different configurations equally, or can be overcome with reasonable cost.

Traverse delay is defined as:

$$T_{tv} = 2 \cdot \left\{ \frac{L_r}{V_r} + \frac{L_l}{V_l} + N_{rt} \cdot T_{rt} + N_{rl} \cdot T_{rl} \right\} \quad (29)$$

where L_r is the distance of radio transmission, which is the distance between the MS and the BS it currently uses, V_r is the propagation speed of radio, which equals to the speed of light, L_l is the total length of wireline transmission in the mesh network, V_l is the propagation speed in wireline, which is approximately two thirds of the speed of light, N_{rt} is the number of nodes along the transmission path in the mesh network, T_{rt} is the average waiting time per node in the mesh network, which includes nodal processing delay, queuing delay, and transmission delay, N_{rl} is the number of servers which are receiving the snapshot and relaying data to/from the VM server, and T_{rl} is the processing and relay time per server in the hand-off chain.

E. Hand-off Duration

Whenever a VM-level hand-off occurs, i.e., an MS detects that it's out of the range of the original BS and the nearest BS belongs to another LSG at the latest update, we set up an anticipated hand-off end time by adding hand-off duration to the current time. The hand-off duration is given by the following equation:

$$T_{ho} = T_x + \frac{L_s}{V_l} + N_s \cdot T_{rt} \quad (30)$$

where T_x is the total time to deliver every bit of a snapshot to media, which is the summation of queuing delay, processing delay, and transmission delay of the snapshot, which is proportional to the size of the snapshot, L_s is the total transmission distance between the current and the next VM servers, and N_s is the number of nodes between two neighboring servers, which always equals to $2N + 1$ in this case.

F. Update Time Points and Cost Charging

Updates occur for two reasons: a hand-off is completed, or an MS reaches an update position. At each update time point, T_{tv} and transaction counts are updated concurrently.

Whenever a position update comes at T_{now}, all hand-off end times registered in queue earlier than T_{now} have to be treated as update time points according to the algorithm described below:

1) Define T_n as the n^{th} earliest hand-off end time in queue, L_{sn} as the total transmission distance between servers corresponding to the n^{th} earliest hand-off in queue, L_r, L_l, N_{rt}, and N_{rl} are the current cost parameters calculated by the MS's current position and hand-off status, and T_{last} as the previous update time.
2) If $T_{now} > T_0$, insert an update time point at T_0, calculate the transaction counts by the Poisson process given user input rate λ and time duration $(T_0 - T_{last})$, set $T_{last} = T_0$, subtract N_{rl} by one, subtract N_{rt} by $\{2N + 1\}$, subtract L_l by L_{s0}, update T_{tv} according to the new parameters, and remove T_0 and corresponding L_{s0} from the queues.
3) Redo step 2 until $T_{now} < T_0$ or the queue is emptied.
4) Calculate the transaction counts by the Poisson process given λ and time duration $(T_{now} - T_{last})$, update T_{tv} according to the new parameters, and set new $T_{last} = T_{now}$.

As specified in UMTS urban mobility model, we update the MSs' positions every 5 meters. Since a hand-off may occur at the same time, we have to handle the extra cost brought by it as well. When a new hand-off occurs with a position update at current time T_{now} while the previous update time is T_{last}, and every hand-off end time earlier than T_{now} is already treated with the above algorithm, we use another algorithm to update cost parameters, which is described below:

1) Register the new hand-off end time and the corresponding L_s in the queue.
2) Increment N_{rl} by one.
3) N_{rt} is recalculated by the MS's current position and added by $\{N_{rl} \cdot (2N + 1)\}$.
4) Let L_l equals to the summation of all L_s's in queue.
5) T_{tv} is then updated accordingly.
6) The transaction counts are calculated by the Poisson process given λ and time duration $(T_{now} - T_{last})$, and then set new $T_{last} = T_{now}$ for the next update.

Every transaction in an update interval is charged with identical T_{tv}. Note that T_{tv} updated at a time point T is applied to the transactions occur *after* T, while transaction counts calculated at T are placed in the time interval ended at T. Although technically we can create a continuous T_{tv} function and integrate it in each update interval to derive a slightly more accurate T_{tv}, it is unnecessarily complex since T_{tv} variation is negligible within the 5 meters (or less) long path.

G. Traverse Time Accounting

The average T_{tv} per transaction is calculated at the end of 100,000 independent simulations, each lasts 86,400 seconds (one day). The simulation results of variable N, T_{rt}, T_{rl}, T_x, and λ, are presented in the following section.

H. Simulation Results

We first simulate how the size of LSAs affects T_{tv} given *nominal* parameters, which are $T_{rt} = 20ms$, $T_{rl} = 500ms$, $T_x = 600s$, and $\lambda = 1.0$. The simulation result is shown in Figure 19.

As we can see in Figure 19, T_{tv} is high in small LSA configurations due to the higher hand-off occurrence rate. As N increases, T_{tv} first descends, levels for a range of N's, and then linearly ascends. The descending for low N's is due to the reduction of hand-off occurrence. The smooth ascending for higher N's is caused by the higher average number of the nodes along the backhaul route and longer average transmission distance while the hand-off occurrence rate is too low to matter. The flat bottom in between is the result of

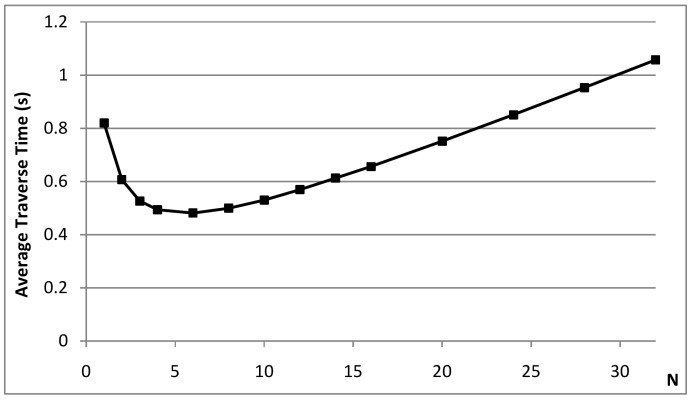

Fig. 19. Simulated T_{tv} of different N given $T_{rl} = 0.5s$ $T_{rt} = 20ms$, and $T_x = 600s$, $\lambda = 1.0$.

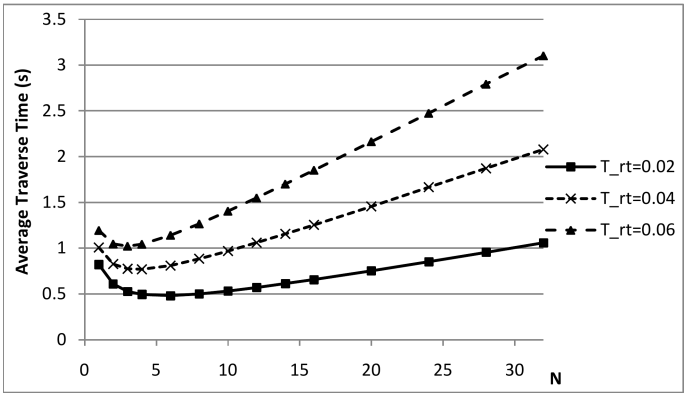

Fig. 21. Simulated T_{tv} given $T_{rt} = 20ms$, $40ms$, $60ms$ and $T_{rl} = 500ms$, $T_x = 600s$, $\lambda = 1.0$.

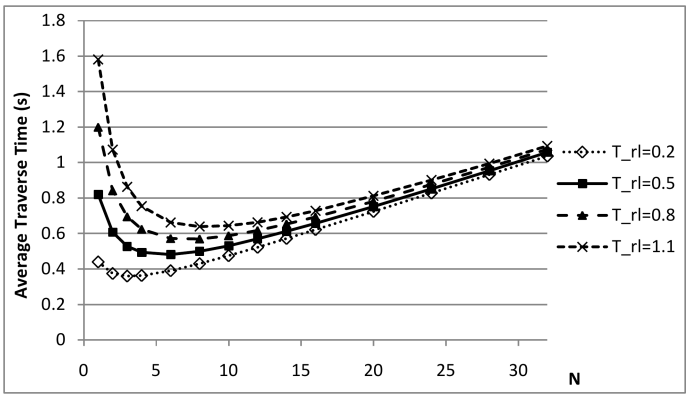

Fig. 20. Simulated T_{tv} given $T_{rl} = 0.2s, 0.5s, 0.8s, 1.1s$ and $T_{rt} = 20ms$, $T_x = 600s$, $\lambda = 1.0$.

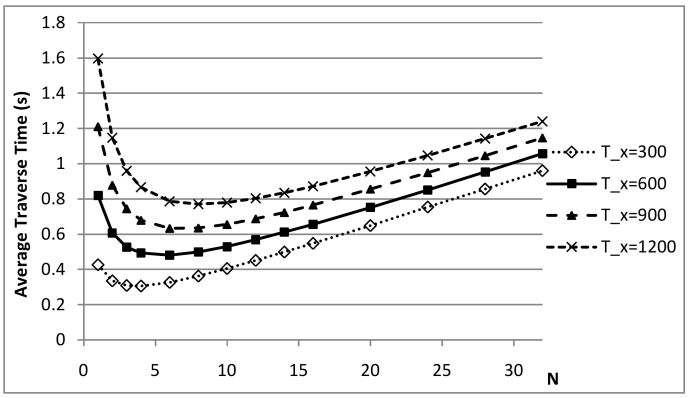

Fig. 22. Simulated T_{tv} given $T_x = 300s$, $600s$, $900s$, $1200s$ and $T_{rt} = 20ms$, $T_{rl} = 0.5s$, $\lambda = 1.0$.

the two effects competing with each other. We can conclude that setting $N = 10$ in this case is optimal in reducing average T_{tv} and keeping the total number of the servers low, which also means lower deployment and maintenance cost.

Since the above conclusion is only applicable in this set of parameters, we adjust each parameter in the nominal set to see how it affects T_{tv} as a function of N in the following subsections.

1) Effect of T_{rl}: T_{rl} is the cost that only applies in hand-offs. We set T_{rl} to $200ms$, $800ms$, and $1,100ms$, to see how it affects T_{tv}. The simulated T_{tv} as a function of N and T_{rl} given $T_{rt} = 20ms$, $T_x = 600s$, $\lambda = 1.0$ is shown in Figure 20.

As we can see in Figure 20, higher T_{rl} significantly increases T_{tv} in small LSA configurations. As N increases, T_{tv} given different T_{rl}'s has a tendency to converge together since the hand-off occurrence rate is dramatically reduced and thus renders the effect of T_{rl} insignificant.

2) Effect of T_{rt}: Unlike T_{rl}, T_{rt} affects both hand-offs and normal transactions since higher T_{rt} amplifies the influence of transmission distance. The simulated T_{tv} as a function of N and T_{rt} given $T_{rl} = 0.5s$, $T_x = 600s$, $\lambda = 1.0$ are shown in Figure 21.

Figure 21 shows the comparison of T_{tv}'s as functions of N given $T_{rt} = 20ms$, $40ms$, and $60ms$. We can easily figure out that as T_{rt} increases, not only T_{tv} increases, but it also increases more sharply for higher N and thus compresses the optimal range of N since higher T_{rt} increases the communication cost per transmission distance in the mesh network. In larger LSA configurations, although hand-offs rarely occurs and thus related cost is minimized, the inner-LSA transmission cost increases more significantly due to the higher nodal cost T_{rt}.

3) Effect of T_x: T_x affects the cost only in hand-offs. Higher T_x may mean larger synchronization data, longer hand-off initialization time, or longer queuing delay. How T_x affects T_{tv} is represented in Figure 22.

Since T_x is the dominant factor of each hand-off's duration, increasing T_x fairly increases the proportion of the transactions occurred during hand-offs for every N. It is why T_{tv}'s as functions of N given different T_x's are virtually parallel to each other and show little tendency to converge as N increases.

4) Effect of λ: Although not being an intuitive factor, we still simulate T_{tv}'s as functions of N given different user input rates λ. The simulated T_{tv}'s given $\lambda = 0.33$, 0.5, and 1.0 inputs per second are almost identical. To visualize the differences, the normalized simulation results are compared in Figure 23.

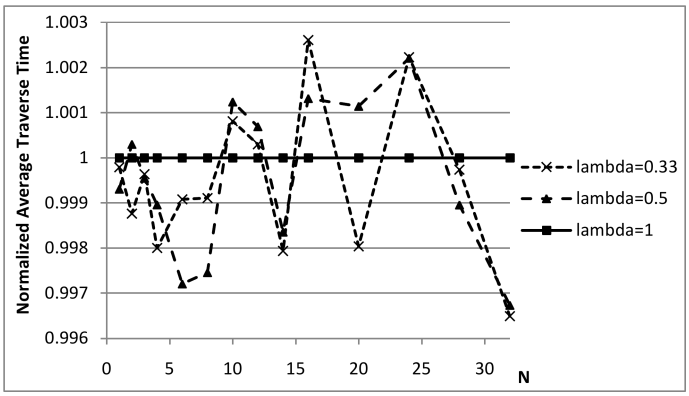

Fig. 23. Normalized simulation results given $\lambda = 0.33$, 0.5, 1.0 and $T_{rt} = 20ms$, $T_{rl} = 0.5s$, $T_x = 600s$.

As we can see in Figure 23, there is no difference induces by adjusting λ per se in statistical view. We should keep in mind, however, that the user experience and the maximum tolerable response delay depend on the interactivity of the application software.

VII. Performance Evaluation Using the UMTS Rural Mobility Model

In the previous section, we used UMTS's urban mobility model to empirically establish the correlations between the performance and the size of each local service area and the capabilities of the network infrastructure. In this section, we employ the Vehicular test environment, also known as the rural vehicular mobility model, of the UMTS document [15] to complete the performance evaluation of the proposed configuration and protocol. The other test environment, that is, the Indoor Office environment described in the UMTS document, will not be discussed in this paper since the communication distance varies relatively small. The Indoor Office environment is more relevant to the radio and baseband design, which is out of our scope. In the UMTS rural vehicular mobility model, BSs are sparsely but optimally placed, MSs move faster and more freely, and the hand-off behavior among base stations is different as well. Although the simulation program in the UMTS rural vehicular model is significantly different from the one presented in the previous section, the concept of the indefinite simulation is retained.

A. UMTS Vehicular Mobility Model

As shown in Figure 24, the UMTS rural vehicular test environment is a plain with no physical obstacle. Each MS's speed is fixed at 120 km/h. Each MS's moving direction is allowed to change up to 45° left or right every 20 meters with 20% chance. All MSs are initially uniformly distributed on the plain.

The BSs in the UMTS rural vehicular test environment are located at the dark grey dots in Figure 24. Each BS has three directional antennae to serve tri-sectored cells. Each cell is assumed to be a hexagon and seamlessly tiles with each other. Each cell's radius R is either 2,000 meters (for services up to 144kbit/s) or 500 meters (for services above 144kbit/s). Therefore, the minimum distance between two BSs can be 6 km or 1.5 km, respectively.

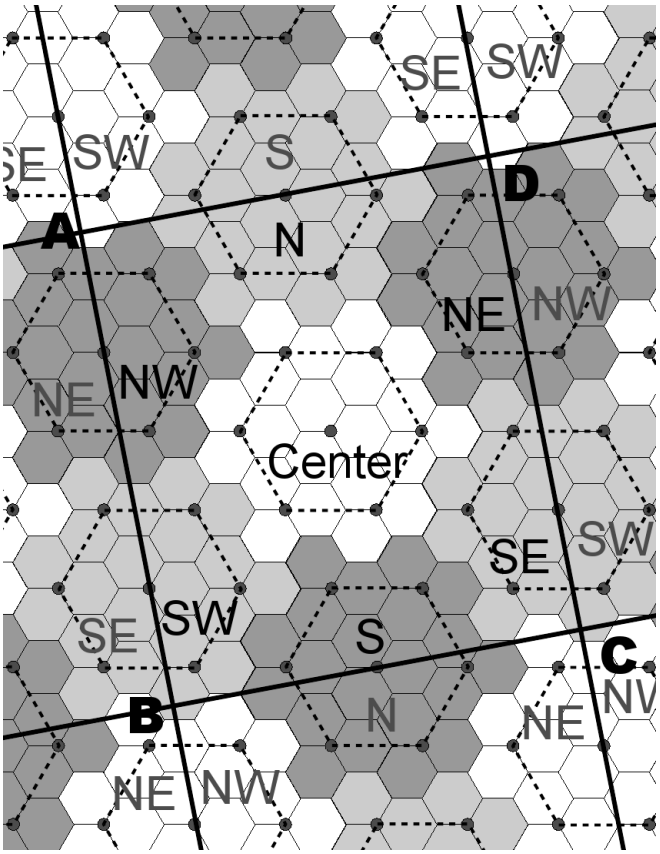

Fig. 24. The UMTS rural vehicular test environment with LSA arrangement.

The original UMTS mobility model generates discontinuities on the boundaries of the test area. We consequently add some special traffic rules, known as *portals*, to eliminate the boundary discontinuities and allow the interaction among LSAs to be simulated and observed for an indefinite period of time. The characteristics of the portals will be detailed in the next section.

B. Möbius County

What interests us is the geographical relation between the service facilities and the MSs' moving space. As the method we conducted in the previous section, the first step is to define a sample area which can represent all the geographical characteristics of service infrastructure we need. We first group the BSs in Figure 24 to form approximately hexagon-shaped LSAs which are optimized in both coverage and average transmission distance by deploying servers at the centers. As the urban counterpart, i.e., Möbius City, in the previous section, the sample area should include one complete LSA in the center and six neighboring halves. Given R and N, the number of the BS intervals per LSA's edge, if we align the origin to the server of an LSG, we define the Parallelogram ABCD surrounded by four straight lines, which are:

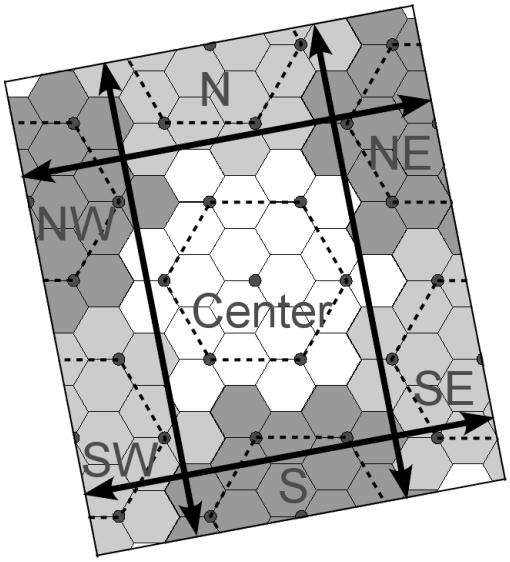

Fig. 25. Möbius County map with teleporting directions.

1) $\sqrt{3}x - 3(2N+1)y = -6\sqrt{3}R(3N^2+3N+1)$ on the north,
2) $\sqrt{3}x - 3(2N+1)y = 6\sqrt{3}R(3N^2+3N+1)$ on the south,
3) $\sqrt{3}(2N+1)x + y = -3\sqrt{3}R(3N^2+3N+1)$ on the west,
4) and $\sqrt{3}(2N+1)x + y = 3\sqrt{3}R(3N^2+3N+1)$ on the east.

as the sample area of our best interest. We can, therefore, crop out Parallelogram ABCD in Figure 24 as our test area, where we call *Möbius County* as shown in Figure 25, to represent every identical piece comprises the indefinite large test area.

Like Möbius City, assigning four logical LSGs in Möbius County is sufficient to figure out when, where, and how frequently an MS moves from one LSA to another. However, to apply the hand-off aborting mechanism, which was disabled in the previous section, we need to distinguish whether an MS is coming back to the LSA it just left or entering the LSA on the opposite side of the one it just crossed. Therefore, we have to assign an additional unique identification for each LSG.

The portals around Möbius County are also similar to those around Möbius City. Whenever an MS is about escaping from Möbius County, the portal teleports it to a proper location at the opposite side so that it reenters Möbius County. Therefore, Möbius County can emulate a limitless test area. Since there is no street structure to align in Möbius County, the rules of the portals are much more simple and straightforward than of Möbius City:

1) For MSs about crossing the north boundary, teleport them to $\big(3R, -3\sqrt{3}R(2N+1)\big)$ from their current locations.
2) For MSs about crossing the south boundary, teleport them to $\big(-3R, 3\sqrt{3}R(2N+1)\big)$ from their current locations.
3) For MSs about crossing the west boundary, teleport them to $\left(-\frac{9R(2N+1)}{2}, -\frac{3\sqrt{3}R}{2}\right)$ from their current locations.
4) For MSs about crossing the east boundary, teleport them to $\left(\frac{9R(2N+1)}{2}, \frac{3\sqrt{3}R}{2}\right)$ from their current locations.

The teleport directions are shown in Figure 25 as well.

The purpose of the portals is to eliminate all discontinuities except the MS's coordinates when it is moving out of the boundary: it keeps the same direction and speed, it associates with the same logical LSG, and preserves the geographical parameters relative to the service group's facilities. Thus, everything interests us is equivalent as the MS moving into an adjacent parallelogram area in a limitless test area.

C. Configuration of Backhaul Network

We assume a mesh-styled backhaul network as we did in the previous section. Therefore, each BS only has direct links to its six neighboring BSs. In the mesh-styled backhaul network, network latency between a BS and the server depends on the number of nodes along the shortest path, the total length of the path, and the relay latency per node. The former two factors are related to the coordinates of the BS and the server, while the last one is varied to simulate different nodal transmission capabilities.

D. Performance Metricand Hand-off Duration

The definition of the traverse delay is identical to the counterpart in the previous section:

$$T_{tv} = 2 \cdot \left\{ \frac{L_r}{V_r} + \frac{L_l}{V_l} + N_{rt} \cdot T_{rt} + N_{rl} \cdot T_{rl} \right\}$$

The hand-off duration is also the same as in the previous section:

$$T_{ho} = T_x + \frac{L_s}{V_l} + N_s \cdot T_{rt}$$

E. Update Time Points and Cost Charging

This part of our simulation program is virtually identical to the counterpart in the previous section. The only differences are: 1) the position update interval is 20 meters instead of 5 meters, 2) the time increment is fixed at 0.6 seconds since each MS's moving speed is always 120 km/h.

F. Traverse Time Accounting

The average T_{tv} per transaction is calculated at the end of 100,000 independent simulations, each lasting 86,400 seconds. The simulation results of variable N, T_{rt}, T_{rl}, T_x, and λ for both $R = 2,000m$ or $500m$, are presented in the following section.

G. Simulation Results

We first simulate how the size of LSAs affects T_{tv} given *nominal* parameters, which are $T_{rt} = 20ms$, $T_{rl} = 500ms$, $T_x = 600s$, and $\lambda = 1.0$. The simulation results of both R settings are shown in Figure 26.

As we can see in Figure 26, both T_{tv}'s bear a strong resemblance in shape to the counterpart in the previous section despite the significantly different mobility models. T_{tv}'s are

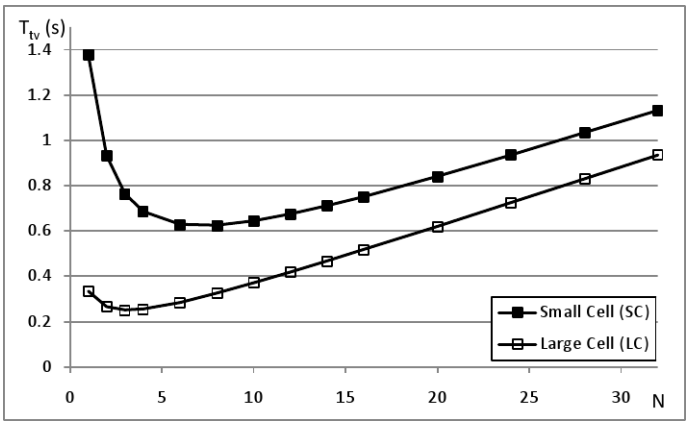

Fig. 26. Simulated T_{tv} of different N of both cell configurations given $T_{rl} = 0.5s$ $T_{rt} = 20ms$, and $T_x = 600s$, $\lambda = 1.0$.

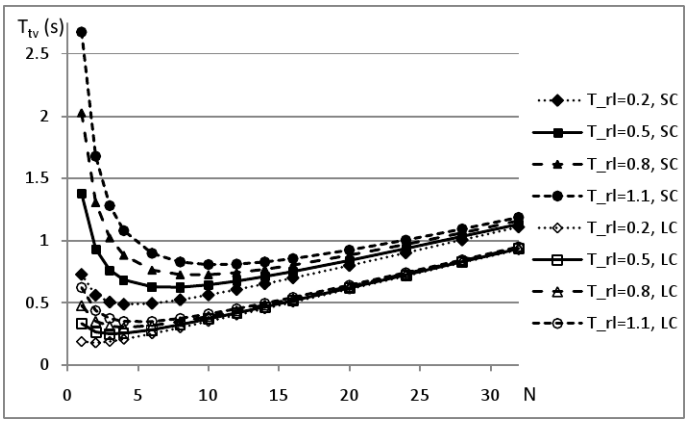

Fig. 27. Simulated T_{tv}'s of both cell configurations given $T_{rl} = 0.2s$, $0.5s$, $0.8s$, $1.1s$ and $T_{rt} = 20ms$, $T_x = 600s$, $\lambda = 1.0$.

high in small LSA configurations due to the higher hand-off occurrence rate. As N increases, T_{tv}'s first descend, level for several N's, and then linearly ascend. The descending for low N's is due to the reduction of hand-off occurrences. The smooth ascending for higher N's is caused by the higher average number of the nodes along the backhaul route and the longer average transmission distance while the hand-off occurrence rate is too low to matter. The flat bottom in between is the result of the two effects competing with each other.

Note although we compare two cell configurations, $R = 2,000m$ and $R = 500m$, in the same figure, each LSA of the former one is in fact 4 times larger than of the latter one. Therefore, each MS encounters much fewer hand-offs in the large cell configuration than in the small cell one. We can also observe slightly steeper ascending for higher N's in the large cell configuration than in the small cell one due to the higher propagation delay brought by the longer wireline and wireless transmission distances.

We can conclude that in this case, setting $N = 4$ for the large cell configuration, and $N = 8$ for the small cell one, are optimal in reducing average T_{tv} and keeping the total number of the servers low, which also means lower deployment and maintenance cost.

Since the above quantitative conclusion is only applicable in this set of parameters, we adjust each parameter in the nominal set and compare the results to see how it affects T_{tv}'s as functions of N in the following subsections.

1) Effect of T_{rl}: T_{rl} only participates in hand-off conditions. In this simulation, we set T_{rl} to $200ms$, $800ms$, and $1,100ms$, and see how it affects both T_{tv}'s. Both simulated T_{tv}'s in large and small cell configurations as functions of N and T_{rl} given $T_{rt} = 20ms$, $T_x = 600s$, $\lambda = 1.0$ are shown in Figure 27.

As we can see in Figure 27, higher T_{rl} significantly increases T_{tv}'s in small LSA configurations due to the higher occurrence rate of hand-offs. As N increases, T_{tv}'s in each cell configuration given different T_{rl}'s have a tendency to converge together since the hand-off occurrence rate is dramatically reduced and thus renders the effect of T_{rl} insignificant. In the large cell configuration, T_{tv}'s converge more significantly and earlier due to the extremely low hand-off occurrence rate.

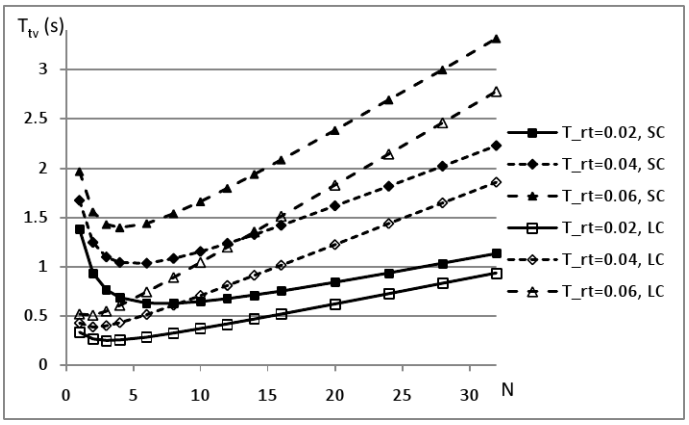

Fig. 28. Simulated T_{tv}'s of both cell configurations given $T_{rt} = 20ms$, $40ms$, $60ms$ and $T_{rl} = 500ms$, $T_x = 600s$, $\lambda = 1.0$.

2) Effect of T_{rt}: Higher T_{rt} amplifies the influence of transmission distance. The simulated T_{tv}'s in both cell configurations as functions of N and T_{rt} given $T_{rl} = 0.5s$, $T_x = 600s$, $\lambda = 1.0$ are shown in Figure 28.

Figure 28 shows the comparison of T_{tv}'s of both cell configurations as functions of N given $T_{rt} = 20ms$, $40ms$, and $60ms$. Besides the resemblance in shape to the counterpart in the previous section, we can also notice that T_{rt} is a more decisive factor for the large cell configuration's performance due to the low hand-off occurrence rate and the long average communication distance in each LSA. Even $N = 1$ can be preferable if T_{rt} is greater than 60ms in the large cell configuration.

3) Effect of T_x: T_x only affects the cost brought by hand-offs. A higher T_x may mean a larger snapshot file, a longer hand-off initialization time, or a longer queuing delay. How T_x affects T_{tv} is represented in Figure 29.

Similar to the counterpart in the previous section, T_{tv}'s of each cell configuration as functions of N given different T_x's are virtually parallel for high N to each other and show very

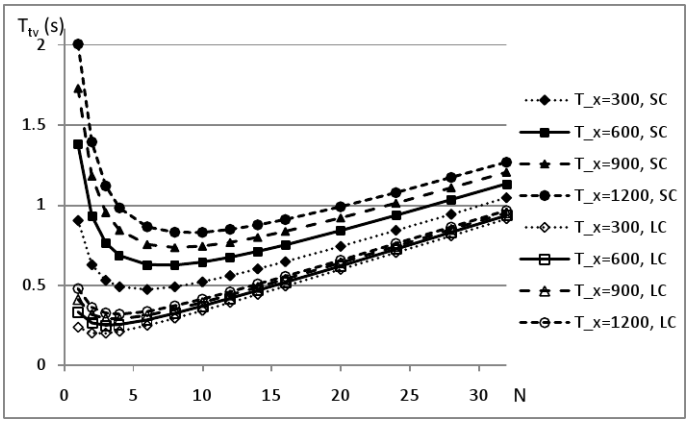

Fig. 29. Simulated T_{tv} of both cell configurations given $T_x = 300s$, $600s$, $900s$, $1,200s$ and $T_{rt} = 20ms$, $T_{rl} = 0.5s$, $\lambda = 1.0$.

little tendency to converge as N increases. However, slightly higher optimal N brought by higher T_x in both configurations is still observable.

4) Effect of λ: Although we have shown that user input rate λ was not a relevant parameter in the previous section, we still simulate T_{tv}'s as functions of N given different user input rates λ in Möbius County. We again confirm that the property doesn't change in the UMTS rural vehicular mobility model.

However, we should keep in mind that the user experience depends more on the interactivity of the application software than on the absolute response latency.

VIII. Conclusion

In this paper, we have first proposed a geographically distributed server arrangement and a hand-off protocol for application virtualization services for mobile users. We have also proposed two analyses to evaluate the impact and the benefit of utilizing the proposed hand-off protocol. And we verify the estimators of probability of a user crossing the borderline by the Monte Carlo experiments and evaluate the accuracies and limitations of both approaches.

After going through the quantitative approaches to compare different server-user configurations, we find out the factors which should be taken into consideration when a service provider plans to launch virtual application services or even virtual desktop services on mobile devices. If they analyze the user behaviors and the application's runtime properties and conclude that their users rarely move, or only move at a low speed, or the data volume required to recreate the runtime environment is relatively small, it is more likely to improve the performance by geographically deploying more servers to cover the whole service area and implement the proposed hand-off protocol. On the other hand, should one or more factors induce a very high hand-off count or overhead, the conventional single server configuration would be preferred.

Following the quantitative approaches, we proposed Möbius City and Möbius County, which are based on the original UMTS urban and rural mobility models but modified to enable MSs to move in the test environment for indefinite period of time without presuming any boundary condition. We simulate the network delay as a result of MSs movements and the occurrences of VM-level hand-offs in Möbius City and Möbius County given variable sizes of LSAs, server relay latencies, routing costs, and transmission delays of snapshots.

By using Möbius City and Möbius County as the test environments, we can evaluate the performance impact and benefit of different sizes of LSAs and infrastructure technologies and capabilities before providing an application virtualization service for mobile computing devices. Möbius City and Möbius County simulations can provide performance previews for planning network infrastructures aim to improve application virtualization services on unknown urban and rural areas, respectively.

Appendix A

The detail derivation of (6):

$$
\begin{aligned}
&\bar{P}_{cross}(L,s,n) \\
&\quad = \frac{2}{3\sqrt{3}L^2}\int_0^s \{n(L-2s)\cdot P_{cross}(d,s)\}\,dd \\
&\qquad + ns^2\left(2-\frac{1}{\sqrt{3}}\right)\cdot\frac{2\bar{\hat{P}}_{cross}}{3\sqrt{3}L^2} \\
&\quad = \frac{2}{3\pi\sqrt{3}L^2}\int_0^s \left\{n(L-2s)\cdot\cos^{-1}\left(\frac{d}{s}\right)\right\}dd \\
&\qquad + \frac{2ns^2(2\sqrt{3}-1)\cdot\bar{\hat{P}}_{cross}}{9L^2} \\
&\quad = \frac{2ns(L-2s)}{3\pi\sqrt{3}L^2}\int_0^1 \cos^{-1}(k)dk \\
&\qquad + \frac{2ns^2(2\sqrt{3}-1)\cdot\bar{\hat{P}}_{cross}}{9L^2} \\
&\quad = \frac{2ns(L-2s)}{3\pi\sqrt{3}L^2}\cdot\left\{k\cos^{-1}(k)-\sqrt{1-k^2}\right\}\Big|_0^1 \\
&\qquad + \frac{2ns^2(2\sqrt{3}-1)\cdot\bar{\hat{P}}_{cross}}{9L^2} \\
&\quad = \frac{2ns(L-2s)}{3\pi\sqrt{3}L^2} + \frac{2ns^2(2\sqrt{3}-1)\cdot\bar{\hat{P}}_{cross}}{9L^2}
\end{aligned}
$$

Appendix B

We assume an MS is at the edge of a service area during hand-offs. Due to the symmetry of hexagons, the average distance from an arbitrary point at the edge of a hexagon to its center is equivalent to the average distance from the 30-degree vertex to an arbitrary point along the opposite leg as shown

in Figure 4.

$$\int_0^{\frac{L}{2}} \left\{ \sqrt{\frac{3L^2}{4} + h^2} \right\} dh$$
$$= \left\{ \frac{h}{2}\sqrt{\frac{3L^2}{4} + h^2} + \frac{3L^2}{8} \ln \left| h + \sqrt{\frac{3L^2}{4} + h^2} \right| \right\} \Bigg|_0^{\frac{L}{2}}$$
$$= \frac{L^2}{4} + \frac{3L^2 \ln(3)}{16}$$

By averaging the result above along the leg, the average transmission distance for an MS at the edge of a service area is:

$$\frac{\frac{L^2}{4} + \frac{3L^2 \ln(3)}{16}}{\frac{L}{2}} = \frac{L}{2} + \frac{3L \ln(3)}{8}$$

Since propagation delay is proportional to the transmission distance, T_{lmax} can be presented in terms of T_{ls} and L:

$$\frac{T_{lmax}}{T_{ls}} = \frac{\frac{L}{2} + \frac{3L \ln(3)}{8}}{\sqrt{3}L}$$
$$T_{lmax} = \left\{ \frac{1}{2\sqrt{3}} + \frac{3\ln(3)}{8\sqrt{3}} \right\} T_{ls}$$

APPENDIX C

The detail derivation of (16):

$$\frac{T_l}{\sqrt{12}} \left\{ 1 + P_{HO}^C \left\{ \frac{12\sqrt{3} + 2 + 6\ln(3)}{4 + 3\ln(3)} \right\} \right\}$$
$$< \frac{T_l}{\sqrt{7}} \left\{ 1 + P_{HO}^B \left\{ \frac{12\sqrt{3} + 2 + 6\ln(3)}{4 + 3\ln(3)} \right\} \right\}$$
$$\sqrt{7}\{1 + P_{HO}^C \cdot k\} < \sqrt{12}\{1 + P_{HO}^B \cdot k\}$$
$$k\{\sqrt{7}P_{HO}^C - \sqrt{12}P_{HO}^B\} < \sqrt{12} - \sqrt{7}$$
$$k \Bigg\{ \sqrt{7} \cdot \frac{2\sqrt{3}E_C(n)\left(\frac{\sqrt{12}s}{L}\right) \cdot \left\{ \frac{D_{sync}}{BW_s} + T_{ls}^C \right\}}{2\sqrt{3}E_C(n)\left(\frac{\sqrt{12}s}{L}\right) \cdot \left\{ \frac{D_{sync}}{BW_s} + T_{ls}^C \right\} + 9\pi\alpha_C}$$
$$- \sqrt{12} \cdot \frac{2\sqrt{3}E_B(n)\left(\frac{\sqrt{7}s}{L}\right) \cdot \left\{ \frac{D_{sync}}{BW_s} + T_{ls}^B \right\}}{2\sqrt{3}E_B(n)\left(\frac{\sqrt{7}s}{L}\right) \cdot \left\{ \frac{D_{sync}}{BW_s} + T_{ls}^B \right\} + 9\pi\alpha_B} \Bigg\}$$
$$< \sqrt{12} - \sqrt{7}$$
$$\frac{8\sqrt{21}\left(\frac{\sqrt{12}s}{L}\right) \cdot \left\{ \frac{D_{sync}}{BW_s} + \sqrt{\frac{7}{12}} T_{ls}^B \right\}}{8\sqrt{3}\left(\frac{\sqrt{12}s}{L}\right) \cdot \left\{ \frac{D_{sync}}{BW_s} + \sqrt{\frac{7}{12}} T_{ls}^B \right\} + 9\pi\alpha_C}$$
$$- \frac{\frac{96}{7}\left(\frac{\sqrt{7}s}{L}\right) \cdot \left\{ \frac{D_{sync}}{BW_s} + T_{ls}^B \right\}}{\frac{16\sqrt{3}}{7}\left(\frac{\sqrt{7}s}{L}\right) \cdot \left\{ \frac{D_{sync}}{BW_s} + T_{ls}^B \right\} + 3\pi\alpha_B}$$
$$< \frac{(\sqrt{12} - \sqrt{7})(4 + 3\ln(3))}{12\sqrt{3} + 2 + 6\ln(3)}$$

where

$$k = \frac{12\sqrt{3} + 2 + 6\ln(3)}{4 + 3\ln(3)}$$

REFERENCES

[1] Joeng Kim, Ricardo A. Baratto, and Jason Nieh, An Application Streaming Service for Mobile Handheld Devices, *SCC'06 IEEE International Conference on Services Computing*, Sept. 2006, pp. 323-326.

[2] VMware Inc., "VMware ThinApp Agentless Application Virtualization Overview."

[3] Ana Fernandez Vilas et al., Providing Web Services over DVB-H: Mobile Web Services, *IEEE Transactions on Consumer Electronics*, Vol. 53, No, 2, May 2007, pp. 644-652.

[4] Apple Inc., "App Store Review Guidelines for iOS apps," 2.7 and 2.8, http://developer.apple.com/appstore/guidelines.html, Retrieved 9 Sep. 2010.

[5] Google Inc., "Android Market Developer Distribution Agreement," 4.5, http://www.android.com/us/developer-distribution-agreement.html, Retrieved 22 Feb. 2011.

[6] VMware Inc., "VMware MVP (Mobile Virtualization Platform)," http://www.vmware.com/products/mobile/overview.html, Retreived 7 Aug. 2011.

[7] Chung-Ping Hung and Paul S. Min, Infrastructure Arrangement for Application Virtualization Services, *The 9th International Information and Telecommunication Technologies Symposium (I2TS 2010)*, 2010, pp. 78-85.

[8] Chung-Ping Hung and Paul S. Min, Service area optimization for application virtualization using UMTS mobility model, *International Conference on Internet Computing (ICOMP 2011)*, 2011, pp. 128-134.

[9] Chung-Ping Hung and Paul S. Min, Performance evaluation of distributed application virtualization services using the UMTS mobility model, *The First International Conference on Mobile Services, Resources, and Users (MOBILITY 2011)*, 2011, pp. 83-89.

[10] Marcin Bienkowski et al., Competitive Analysis for Service Migration in VNets, in *Proc. 2nd ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures*, 2010, pp. 17-24.

[11] Dushyant Arora, Anja Feldmann, Gregor Schaffrath, and Stefan Schmid, On the benefit of virtualization: strategies for flexible server allocation, *Hot-ICE'11 Proceedings of the 11th USENIX conference on Hot topics in management of internet, cloud, and enterprise networks and services*, 2011.

[12] L. Peter Deutsch and B. W. Lampson, *SDS 930 Time-sharing System Preliminary Reference Manual*, Doc. 30.10.10, Project Genie, Univ. Cal. at Berkeley, April 1965.

[13] VMware Inc., "Virtual Desktop Infrastructure."

[14] R. Buckminster Fuller, *Synergetics: explorations in the geometry of thinking*, Macmillan Publishing Company, 1975.

[15] ETSI. Universal Mobile Telecommunications System (UMTS); selection procedures for the choice of radio transmission technologies of the UMTS (UMTS 30.03, version 3.2.0), Technical report, European Telecommincation Standards Institute, Apr. 1998.

[16] H. Boche and E. Jugl, Extension of ETSI's Mobility Models for UMTS in Order to Get More Realistic Results, *Proc. UMTS Workshop*, Günzburg, Germany, Nov. 1998.